commodore Magazine

AÑO I - Núm. 4 - Junio 1984 - 250 Ptas.

REVISTA INDEPENDIENTE PARA USUARIOS



CONCURSO CALC RESULT

UN PREMIO DE 80.000 PTAS. EN MATERIAL COMMODORE

a elegir por el ganador, espera a la aplicación más original y útil que envíen nuestros lectores.

Las hojas de trabajo ofrecen elevadas posibilidades de utilización. Sin embargo, creemos que en nuestro país todavía no se ha captado el enorme potencial que ofrecen. Por todo ello, "COMMODORE MAGAZINE" convoca un concurso de aplicaciones desarrolladas bajo CALC RESULT, sean financieras, dietéticas, etc...

■ El fallo del concurso se publicará en el número 4 de nuestra revista.

■ Para participar se enviará una detallada descripción de los objetivos que pretende la aplicación y la metodología utilizada.

■ El premio podrá ser declarado desierto, prorrogando en 2 meses la aceptación de nuevas aplicaciones en tal caso.

■ La aplicación premiada será publicada en forma de artículo. En caso de empate, el premio se dividirá en partes iguales entre los ganadores.

■ Los miembros del Jurado serán elegidos por "Commodore Magazine" entre cualificados profesionales que evaluarán la utilidad de la aplicación, siendo su decisión inapelable.

■ La fecha tope para la admisión de aplicaciones es el 1 de mayo de 1984. Todas las aplicaciones deben enviarse a:



Calc Result
"Commodore Magazine",
C/Bravo Murillo, 377. Madrid -20

commodore Magazine

Sumario

Commodore Magazine es una publicación de Ediciones y Suscripciones S.A., C/Bravo Murillo, 377 - Madrid 20, Tel. (91) 733 74 13 / 47 / 63 / 97.
REDACCION
Director:
Alejandro Diges.
Colaboradores:
Anibal Pardo.
Gumersindo García.
Roberto Menéndez.
Simeón Cruz.
Miguel Angel de Frutos.
Manuel Arias.
Diseño:
Ricardo Segura.
EDITORIAL
Presidente:
Fernando Bolín.
Director Editorial:
Norberto Gallego.
ADMINISTRACION
Gerente de Circulación y Ventas: Luis Carrero.
Suscripciones: Antonio Zurdo.
Producción:
Miguel Onieva.
Publicidad Madrid:
Roberto Rodriguez.
Bravo Murillo, 377.
Madrid - 20.
Tel. (91) 733 74 13.
Publicidad Barcelona:
Pelayo, 12.
Tel. (93) 301 47 00, Ext. 27
Distribuye: SGEL. Avda. Valdelaparra s/n, Alcobendas, Madrid.
Imprime: Novograph S.A., Ctra. de Irún, Km 12.450 Madrid.
Fotomecánica: Karmat. Pantoja, 10, Madrid.
Depósito Legal: M-6622-1984
Solicitado control de OJD.

Año 1
Num. 4

Editorial

Los recientes cambios en la directiva de **Commodore International** no parece que vayan a producir sustanciales desvios en la forma de operar de la compañía. Es más, el usuario se beneficiará de un aumento en la gama de productos ofertados. Ya se habla de una nueva familia de ordenadores domésticos, la **Commodore Plus 4,** heredera directa del aun no comercializado **264.**

Recientemente la prensa económica se hacía eco de una serie de cambios en la plantilla de la compañía. En unos casos se ha debido a despidos, en otros a abandonos voluntarios. Se habla de hasta ocho altos ejecutivos, entre los que se incluyen cuatro vicepresidentes. Incluso parece ser que se ha marchado el propio hijo de **Jack Tramiel,** fundador de la compañía, **Sam.** No obstante, también hay acontecimientos interesantes. Los próximos días siete a nueve de junio va a celebrarse en Londres el **Quinto International Commodore Computer Show,** feria a la que acuden todos aquellos que tienen algo que ofertar a los usuarios de la marca. Allí estará **Commodore Magazine,** como mero espectador, para contaros todo lo que de interés veamos. Los organizadores aseveran que podremos ver el **C-16,** que tuvimos ocasión de conocer en la **Feria de Hannover,** y los ordenadores **Commodore Plus 4,** incluso un nuevo conjunto de periféricos. El Show se divide en dos grandes secciones, la destinada a los sistemas destinados a la empresa y los negocios, y la informática doméstica y recreativa. También está anunciado que concurra **Patrick Moore,** para mostrarnos un sensacional programa de Astronomía para el **64.** La red diseñada para unir varios de estos modelos, la **Compunet** estará igualmente presente. Se anuncia una exposición de Arte por Ordenador, trataremos de conseguir copias de algunos originales para publicación en nuestras páginas.

En el presente número os describimos un juego que servirá para hacernos pasar agradables horas, es el **International Soccer,** uno de los juegos más apasionantes que hemos visto para ordenadores personales.

El ordenador transportable **SX-64** es otro de los temas de portada elegidos, aunque en esencia es prácticamente un ***64,*** merecía la pena describir sus peculiaridades.

La posibilidad de interfasar el **Vic-20** con una impresora estándar RS232 nos ha parecido lo suficientemente atractiva, como para dedicarle la primera parte de una serie de dos artículos, cuya terminación publicaremos en el siguiente mes. El número se completa con la segunda parte de la serie "La otra forma de leer el Manual del 64" y, por supuesto, las secciones fijas de programas y concurso.

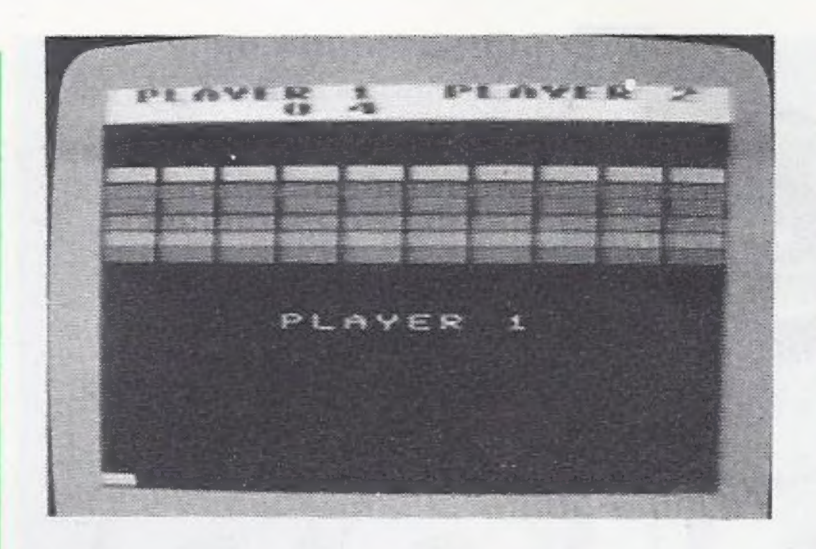
Claves para introducir en el ordenador los programas que aparecen en la revista.

COMO SE VE	COMO SE TECLEA	EFECTO CONSEGUIDO
	SHIFT + CLR	(LIMPIA PANTALLA)
	CLR	(HOME)
	CRSR	(CURSOR ABAJO)
	SHIFT + CRSR	(CURSOR ARRIBA)
	CRSR	(CURSOR DERECHA)
	SHIFT + CRSR	(CURSOR IZQUIERDA)
	CTRL + 1	(NEGRO)
	CTRL + 2	(BLANCO)
	CTRL + 3	(ROJO)
	CTRL + 4	(CIAN)
	CTRL + 5	(VIOLETA)
	CTRL + 6	(VERDE)
	CTRL + 7	(AZUL)
	CTRL + 8	(AMARILLO)
	CTRL + 9	(CARACTER INVERSO)
	CTRL + 0	(CARACTER NORMAL)

PROGRAMA: SUPER SMASH
TIPO: JUEGO
DISTRIBUIDOR: MICROELECTRONICA Y CONTROL
FORMATO: CARTUCHO
COMPUTADOR: COMMODORE 64 CON PADDLES

Aunque en la portada de la caja, en la que se presenta el juego, aparezca un atlético tenista insinuando un gran juego con gráficos increíbles, el juego en sí es el antiguo (si antiguo se le puede llamar a un juego con 6 ó 7 años) desenladrillador que no hace demasiado tiempo jugamos en los bares. Básicamente consiste en un muro de ladrillos que tenemos que derribar usando una raqueta y una pelota.

El cartucho se conecta bien (no se olvide de apagar antes el ordenador si no quiere jugar al bombero-apaga-ordenadores) y el juego empieza automáticamente al encender la máquina, ofreciéndonos un menú con varios tipos de juego y la opción de uno o dos



jugadores. Los tres juegos disponibles son: **Super Smash,** que es el juego descrito anteriormente, en él hay seis filas de ladrillos, cada una de un color y tenemos que derribarlos todos. En cualquiera de las tres opciones, cuando lleguemos a la quinta fila, la bola se acelerará y si golpeamos la pared del fondo el tamaño de la raqueta disminuirá a la mitad. Aún así el juego resulta bastante fácil y las cinco pelotas de que disponemos resultan suficientes. El segundo juego es el denominado ESPECIAL, en el cual son tres bloques de ladrillos separados entre sí por huecos. Cuando la pelota se mete entre dos de ellos empieza a rebotar, dándonos una gran cantidad de puntos. La última variante es la PROGRESIVE, similar en forma a la ESPECIAL, pero con la particularidad de que los ladrillos van avanzando poco a poco hacia nosotros. Las tres versiones son muy similares entre sí, con las diferencias ya señaladas, por lo que decir que son tres juegos distintos como viene en el manual es algo exagerado. El juego viene presentado en una atractiva caja con un manual muy extenso (naturalmente en inglés), que aparte de las instrucciones en sí incluye unos consejos prácticos para obtener más puntos. Un defecto de este programa consiste en que utiliza *paddles,* accesorio que casi nadie tiene y que hará que su uso se vea bastante restringido.

PUNTUACION:
ADICCION: 6
PRESENTACION: 7
GRAFICOS: 3
ACCION: 4

PROGRAMA: CUDDLY CUBURT
TIPO: JUEGO
DISTRIBUIDOR: INDESCOMP-ABC SOFT
FORMATO: DISCO Y CASSETTE
COMPUTADOR: COMMODORE 64

Este juego sobre el que parece haber un litigio de propiedad, ya que lo distribuyen dos casas distintas, está basado en un *arcade game* (juego de "marcianitos") muy popular en las salas de juegos de Estados Unidos, aunque en España es desconocido, nos estamos refiriendo al **Q-Bert** del que es una copia casi exacta.

Una vez puesto en marcha vemos una pirámide de grandes dados y un curioso muñeco **(Cuddly Cuburt)** que tiene que ir saltando de uno a otro, cada vez que salta encima de uno la parte superior del cubo cambiará a otro color, debiendo hacer que cambie toda la pirámide a un color indicado a la izquierda. La labor se ve complicada por diversos objetos que van apareciendo durante el juego (balones, serpientes) y que si le dan, le matan. Otra cosa que puede suceder es que se caiga por los bordes de la pirámide (que según parece está flotando en el vacío), con lo cual, además de perder una vida perderá todos los cubos que ya hubiese pintado y tendrá que volver a empezar. Para ayudarle en su misión hay dos discos volantes a ambos lados de la pirámide, si salta en ellos le conducirán a la parte superior de la pirámide, salvándole de cualquier peligro inmediato y si además era perseguido por la serpiente, caerá al vacío al intentar seguirle. Tenga cuidado y no malgaste los discos, ya que una vez usados desaparecen, úselos sólo en caso de extrema necesidad. El juego dispone de una gran cantidad de niveles, que se van complicando sucesivamente, después del nivel cinco necesitaremos saltar más de una vez encima de cada cubo para cambiarlo al color que buscamos.

El juego aunque parece simple y sin interés en un principio, luego demuestra ser uno de los más divertidos que existen actualmente, requiere una gran habilidad y control, pues los saltos y desplazamientos se realizan en diagonal y no en vertical y horizontalmente como estamos acostumbrados. Por último, hay que señalar que se puede jugar con *joystick* o con teclado.

PUNTUACION:
ADICCION: 7
PRESENTACION: 6
GRAFICOS: 6
ACCION: 6

Fútbol de salón

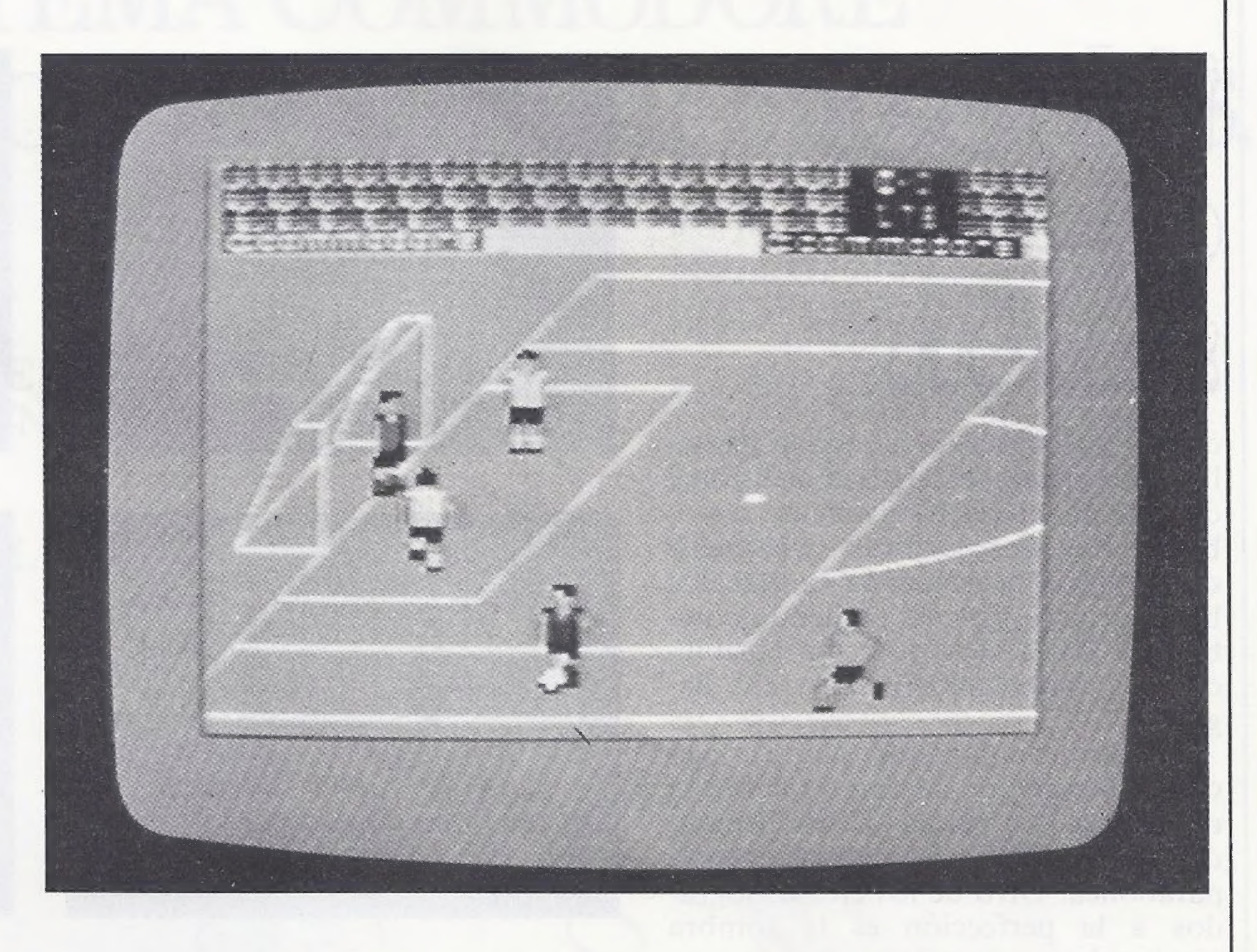
Una tarde lluviosa en la ciudad. Los dos equipos de fútbol con mayor rivalidad deportiva se disponen a salir al terreno de juego. La moneda es lanzada al aire y los hinchas, con su griterío, hacen que el estadio parezca venirse abajo. Sin embargo, nosotros no soportamos las inclemencias meteorológicas, pero los nervios están a flor de piel. El resultado del partido dependerá en gran medida del resultado de nuestro juego.

Aquel viejo sueño, de poder practi-



car el fútbol, sin necesidad de salir de casa, se ha materializado en la realidad. Efectivamente, ya no es necesario más que ser el feliz propietario de un **CBM 64**, para no conformarse simplemente con ver ganar o perder al equipo favorito. A partir de ahora seremos nosotros quienes decidamos su futuro. Se acabaron los inútiles consejos, que a gritos, dábamos a los delanteros, desgañitándonos al comprobar que no podían oírlos.

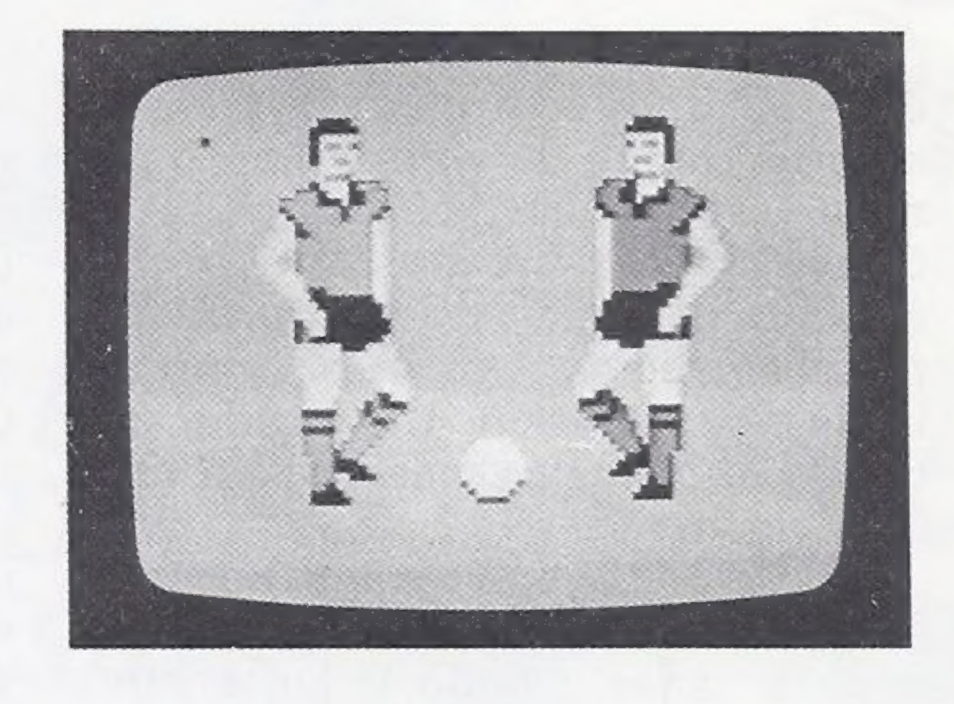
International Soccer es un cartucho de juegos que ha puesto recientemente **Commodore** en circulación. Su calidad gráfica está muy por encima de la media a que estamos acostumbrados. En realidad supera con mucho a los tradicionales juegos de marcianitos. Aunque la visualización en pantalla se efectua, lógicamente, en dos dimensiones, el efecto del movimiento tridimensional está conseguido con gran dosis de realismo.



El juego está contenido en uno de esos tradicionales cartuchos que utiliza **Commodore** para empaquetar sus programas de estado sólido. Se conecta al *port* de expansión, dispuesto en la parte posterior del **64**, teniendo mucho cuidado de que la alimentación está desconectada. De lo contrario se podría dañar la circuitería del ordenador y el cartucho. Una vez hecho esto, se activa el conmutador de alimentación del **64**, apareciendo en pantalla durante bastantes segundos la molesta publicidad del fabricante. Lo que ocurre a continuación, si no se presiona tecla alguna, es que comienza un pequeño turno demostrativo de las capacidades del juego.

Las posibilidades de ese resumen en dos modos: contra el ordenador o contra un segundo oponente humano. El *joystick*, con su pulsador de disparo, es el complemento necesario para evolucionar por el campo. En el segundo caso se precisan dos *joystick*, uno para cada equipo.

Cuando se juega contra el ordenador, se puede elegir el nivel de dificultad, en una escala de 1 a 9. La tecla de función F5 es la encargada de establecerlo. El 9 es el grado de mayor dificultad, el que más emoción imprime al juego. Por el contrario, el 1 es el más simple, que nos servirá para

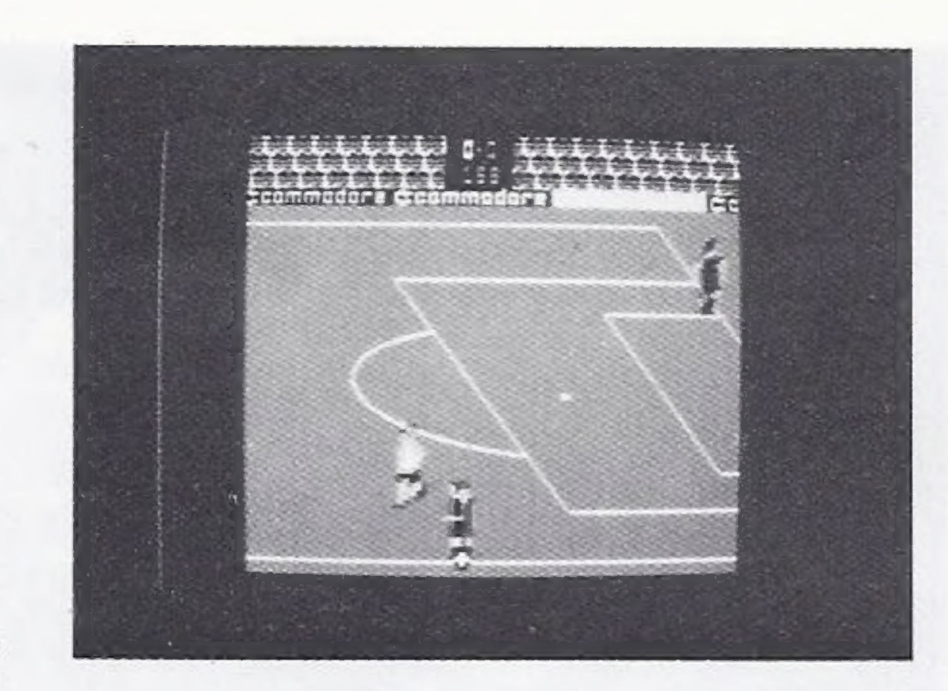


familiarizarnos con las características del juego.

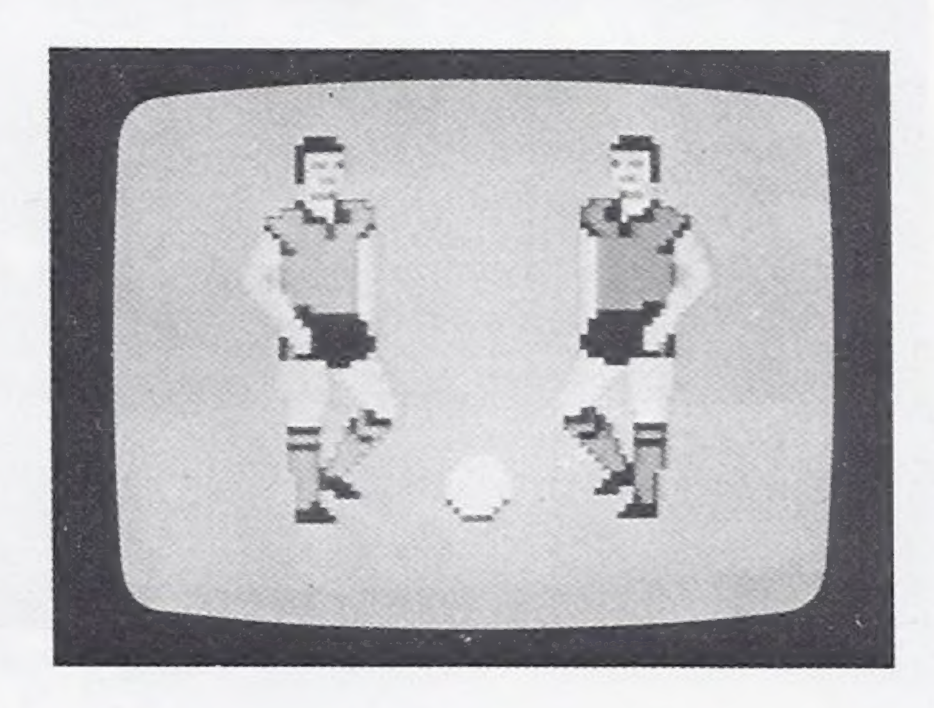
Antes de comenzar existe la posibilidad de elegir el color de la camiseta a defender. Aparece la representación de dos jugadores en tamaño superior al normal. La tecla F1 se utiliza para ir cambiando el color del equipo que juegue a la izquierda de la pantalla, mientras que la F3 lo hace con el equipo de la derecha. Los colores a elegir son seis en total: amarillo, azul, rojo, naranja, blanco y gris. Cuando no se utiliza televisor o monitor de color, puede ser eliminada la señal de color mediante la tecla F7, apareciendo más nítidos los jugadores al disponer solamente de la escala de grises, blanco y negro.

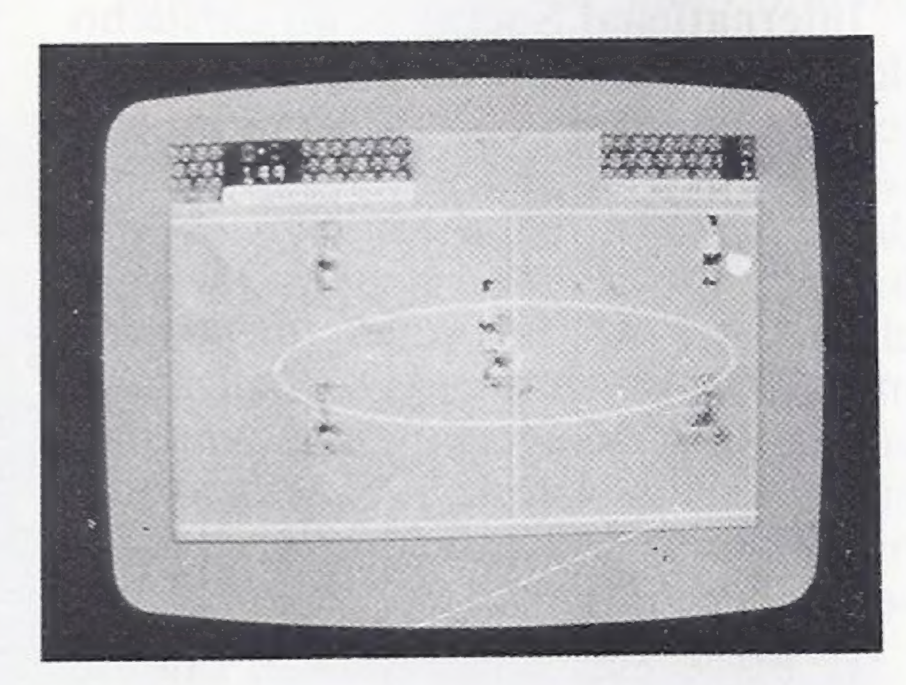
El juego comienza, una vez elegidos los colores, cuando se presiona el botón de disparo del *joystick*. Los jugadores comienzan a salir de los vestuarios, por la parte superior central de la pantalla, al terreno de juego, representado con todo realismo, incluyéndose el verde césped y las líneas de demarcación. Una vez alineados los jugadores para comenzar a jugar, permanecen de pie, disciplinadamente, hasta que el árbitro (que nunca aparece en la pantalla) hace sonar nítidamente su silbato. Este es uno de los momentos de mayor emoción. Cada participante ejerce control absoluto sobre su equipo, con la ayuda del *joystick* correspondiente. Como es obvio, no se puede controlar a todos y cada uno de los jugadores con un solo *joystick*. Por lo tanto, se controla a un jugador. El jugador de cada equipo que está más próximo al balón es el que podemos controlar solidariamente al movimiento de la palanca del *joystick*. Para que resulte fácilmente identificable de quien se trata, el color de la camiseta de ese jugador aparece en el mismo color elegido, pero en tono más claro. De todas maneras, es sorprendente comprobar como el jugador se desplaza en las

Fútbol de salón









cuatro direcciones por toda la pantalla, con lo que el efecto tridimensional de movimiento es francamente asombroso. En otras palabras, se le puede dirigir hacia la izquierda, la derecha, hacia delante y atrás, pudiendo combinarse hasta dos de estas posibilidades para el movimiento en diagonal.

Hacer que el jugador "chute" no es cuestión más que de apretar el botón de disparo. La pelota saldrá disparada, observando su normal trayectoria parabólica. Otro de los efectos logrados a la perfección es la sombra producida por la pelota, cuando vuela, sobre el césped. El efecto de realidad creado es un absoluto logro.

También hay mucho que decir de los jugadores, son verdaderos profesionales, capaces de hacer jugadas y remates de cabeza, *dribblings* y toda una suerte de jugadas de regate del balón. Los *corners* también son lanzados con gran maestría, y que decir de los fuera de banda. Aquí no hay duda, el mejor juez de línea es el propio ordenador, que mantiene en su memoria quien fue el último en tocar el balón.

Los golpes son una auténtica gozada, que como en los partidos reales, desatan el entusiasmo del público. Las gradas, abarrotadas, parecen venirse abajo con cada gol. Puede verse a los aficionados moverse desenfrenadamente, y el *chip* sintetizador de sonido produce una síntesis realista del griterío de la multitud.

Para mayor deportividad de la competición, a este juego no se le ha dotado de faltas o de lanzamiento de almohadillas y botes de refresco al terreno de juego.

Solamente se visualiza un trozo del campo simultáneamente, desplazándose esta porción a medida que el balón avanza *(scrolling).* Si el jugador que podíamos controlar sale fuera de nuestro campo de visión, inmediatamente es uno nuevo el que cambia el color de su camiseta.

El desplazamiento de los jugadores por el campo no se produce siempre a la misma velocidad. Un jugador haciendo *dribbling* va más despacio que otro sin balón.

El portero se entrega igualmente a fondo en su tarea. Resulta gratificante verle saltar cuando le lanzan tremendos chutes. Muy al contrario de lo que pudiera parecer, es posible elaborar estrategias no se trata de un juego que lleva todas sus posibilidades mecánicamente programadas, ni mucho menos. Cuando los dos participantes han desarrollado un determinado grado de maestría con el **International Soccer**, la semejanza con un partido de fútbol televisado es total. Este es uno de esos juegos capaces de crear adicción en los participantes.

Como en los partidos clásicos, aquí también existen dos tiempos de juego, **Commodore** dice en el folleto explicativo que es de 200 unidades de tiempo cada mitad.

El equipo más ofensivo, más experto y con más suerte, logra batir mayor número de veces la portería de su oponente. Claro, es lógico que la pericia del portero tenga mucho que decir. Al fin y al cabo este para los balones por indicación del pulsador de disparo del *joystick* correspondiente.

Una vez concluido el partido, los jugadores del equipo ganador se vuelven a alinear, y una bella damisela vestida de largo les premia con la dorada Copa de Campeones.

En resumen podemos decir que este cartucho exprime al máximo las características gráficas y el control de *sprites* del **Commodore 64**, creando efectos tan realistas que a menudo se toma el juego como algo real.

CONCURSO MENSUAL DE COMMODORE MAGAZINE

SU PROGRAMA PARA CUALQUIER SISTEMA COMMODORE PUEDE HACERLE GANAR 5.000 PTAS.

EL PRESENTE CONCURSO ESTA ABIERTO A TODOS NUESTROS LECTORES Y SU PARTICIPACION E INSCRIPCION ES GRATUITA. LEA LAS BASES DEL CONCURSO

- NO SE ESTABLECEN LIMITACIONES EN CUANTO A EXTENSION, TEMA ELEGIDO O MODELO DE ORDENADOR
- LOS CONCURSANTES DEBERAN ENVIARNOS A LA DIRECCION QUE FIGURA AL PIE, EL CASSETTE O DISKETTE CONTENIENDO EL PROGRAMA, UNA EXPLICACION DEL MISMO Y, AL SER POSIBLE, UN LISTADO EN PAPEL DE IMPRESORA. SE PODRAN ENVIAR TANTOS PROGRAMAS COMO SE DESEE
- LOS PROGRAMAS, PREVIA SELECCION, SERAN PUBLICADOS EN LA REVISTA, OBTENIENDO TODOS ELLOS 5.000 PTAS.
- LA DECISION SOBRE LA PUBLICACION O NO DE UN PROGRAMA CORRESPONDE UNICAMENTE AL JURADO NOMBRADO AL EFECTO POR "COMMODORE MAGAZINE", SIENDO SU FALLO INAPELABLE
- LOS CRITERIOS DE SELECCION SE BASARAN EN LA CREATIVIDAD DEL TEMA ELEGIDO Y LA ORIGINALIDAD Y/O SENCILLEZ EN EL METODO DE PROGRAMACION GLOBAL
- ENVIAR A: CONCURSO COMMODORE MAGAZINE



commodore *Magazine*

C/Bravo Murillo, 377. Madrid -20

Iniciación al lenguaje má

Una de las cosas más antipáticas con las que se encuentra todo aquel que desea familiarizarse con el lenguaje máquina y los microprocesadores es lo aburrido y tedioso de entender y memorizar lo relativo a los modos de direccionamiento.

Si bien los hemos explicado en una ocasión anterior, no está de más redundar en ellos, pero con una óptica distinta.

Lo más importante que aportan los microprocesadores, y por extensión los microordenadores y demás sistemas informáticos, es la gran rapidez con la que pueden ejecutar cálculos. Una de las instrucciones más interesantes es la comparación de datos, y la posterior toma de decisiones en función de ella.

El 6500 dispone de un conjunto de 14 instrucciones relacionadas con la comparación. Mediante ellas se pueden comprobar los valores de los datos contenidos en el acumulador y los registros X e Y del microprocesador.

Cada una de las mencionadas instrucciones compara el contenido del registro elegido con el byte contenido en una dirección de memoria, especificada en el operando de la instrucción. El resultado que produce es la alteración de determinados bits (flags) del registro de estado.

Pasemos a enumerar cuáles son las comparaciones que afectan al acumulador en la figura 1.

En realidad, todas estas instrucciones afectan únicamente a tres bits (flags) del registro de estado, que son el de signo cero y acarreo (N, Z, C).

Analicemos lo que hace el microprocesador cuando compara dos bytess. Realmente, no hace otra cosa que ejecutar una sustracción del byte contenido en el operando al byte contenido en el acumulador. La utilización de CMP afecta a los siguientes flags del registro de estado: Z, que se pone a uno cuando el resultado de la comparación es igual; N se pone a uno o cero, según sea el resultado del bit de mayor peso del byte, y C que se pone a uno cuando el valor contenido en el operando es menor o igual que el acumulador. C se pone a cero si es mayor que el contenido del acumulador. El cuadro mostrado en la figura 2 resume el comportamiento.

La principal diferencia con otras operaciones, tal como pudiera ser la propia resta, es que el contenido del acumulador no resulta alterado como resultado de la comparación. Es más, que el acumulador permanezca inal-

Figura 1.

```
Comparación con el Acumulador:

CMP #n     .... Inmediato.
CMP nn     .... Absoluto.
CMP n      .... Página cero.
CMP (n,X)....  Indexado Indirecto.
CMP (n),Y....  Indirecto Indexado.
CMP n,X    .... Página cero Indexado
                en X.
CMP nn,X   .... Indexado en X.
CMP nn,Y   .... Indexado en Y.

Comparaciones con el registro X:

CPX #n     .... Inmediato.
CPX nn     .... Absoluto.
CPX n      .... Página cero.

Comparación con el registro Y:

CPY #n     .... Inmediato.
CPY nn     .... Absoluto.
CPY n      .... Página cero.
```

...quina Familia 6500

...terado permite un elevado grado de flexibilidad en la programación.

En el cuadro 2 aparece el término "complemento a dos". La mayoría de los microprocesadores, incluyéndose el 6502, utilizan la notación conocida como sistema de numeración con complemento a dos, para representar cifras afectadas de signo. Utilizando números escritos en forma binaria mediante ocho bits, el margen de valores va desde —128 hasta —1 y desde cero hasta +127.

Esto se debe a que el bit de mayor peso del byte (el octavo) es quien establece el signo del número. Si este bit es cero, el número formado por los otros siete bits es positivo, de lo contrario es negativo.

Para obtener el complemento a dos de un número positivo, lo primero que hay que hacer es invertir todos los bits (poner ceros donde había unos y viceversa) y después sumarle uno a este resultado inverso.

El complemento a dos resulta ser muy útil en la aritmética binaria, más adelante veremos detalladamente su utilidad, pero por el momento nos conformaremos con conocer su existencia.

La instrucción de comparación trabaja siempre de la misa forma, independientemente del método de direccionamiento empleado. Sin embargo, sería una operación inútil si no se pudiera hacer nada con los resultados obtenidos con ella.

El 6502 dispone de un conjunto de ocho bifurcaciones condicionales, que son las que indica el cuadro de la figura 3.

El operando de estas instrucciones de bifurcación está compuesto por un solo byte. Puede parecer chocante al compararlo con las instrucciones JMP y JSR, que pueden saltar sin mayor problema a cualquiera de las 64K direcciones de la memoria indicada en los dos bytes que siguen a la instrucción.

Al contrario de las otras instrucciones que implican bifurcación, las ocho especificadas en el cuadro 3 emplean únicamente el modo de direccionamiento relativo, que sólo les permite saltar 128 bytes hacia atrás o 127 hacia delante. Estas cifras y el sentido del salto probablemente nos recuerdan las reflexiones de cifras y signo que veíamos en el complemento a dos. En efecto, se trata de la misma cosa. Cuando utilizamos un ensamblador para desarrollar nuestro programa en lenguaje máquina, es aquel quien calcula el byte correspondiente al salto. Sin embargo, cuando programemos a las bravas, existe un pequeño truco que nos permite calcular el valor del byte. Simplemente contemos hacia atrás o delante el número de bytes que median entre el anterior a la instrucción de salto y la posición final. Si el salto es hacia atrás, comenzaremos con 255 restándole uno por byte. Si es hacia delante se le va sumando uno, partiendo de cero.

Cuando el salto deba ser efectuado a una dirección de memoria situada fuera del alcance de +128 ó —127, la inclusión de una instrucción JMP complementará esta limitación.

La correcta utilización de las instrucciones de comparación y bifurcación son, en gran medida, quienes dotan de su potencia de proceso a los ordenadores, al permitirles tomar "decisiones" en función de los datos de que dispone.

Para finalizar este capítulo, digamos que algunas veces no resulta necesario recurrir a CMP, pues es el resultado de otra operación quien altera los registros necesarios para orientar la bifurcación.

Figura 2.

Comparación	Registro de Estado N	Z	C
Acc, X ó Y < Memoria	1*	0	0
Acc, X ó Y =Memoria	0	1	1
Acc, X ó Y >=Memoria	0*	0	0

* Afectado solamente en el complemento a dos.

Instrucciones de bifurcación:

BCS n..	Bifurcar	si	C=1
BCC n..	Bifurcar	si	C=0
BEQ n..	Bifurcar	si	Z=1
BNE n..	Bifurcar	si	Z=1
BMI n..	Bifurcar	si	N=1
BPL n..	Bifurcar	si	N=0
BVC n..	Bifurcar	si	V=0
BVS n..	Bifurcar	si	V=1

Figura 3.

Ya está a la venta el nº 16



El ordenador



transportable

Un buen día **Commodore** se decidió a poner el **CBM 64**, la unidad de *diskettes* y el monitor de color en un bonito y reducido paquete. El resultado fue el modelo **SX-64 Executive**, que como su propio nombre indica, fue orientado a su utilización por parte de los ejecutivos. Su aspecto general recuerda bastante al otrora popular **Osborne 1**. Su peso, sin embargo, es algo menor, alrededor de unos 10 kilos.

El microprocesador utilizado sigue siendo el 6510, igual que para el **64** normal, siendo ambos totalmente compatibles en lo que a *software* se refiere.

La memoria RAM central es de 64 Kbytes, expansibles hasta 256 Kbytes. De los 64 K estándar, 38 K son totalmente disponibles para ser utilizados por los programas en BASIC.

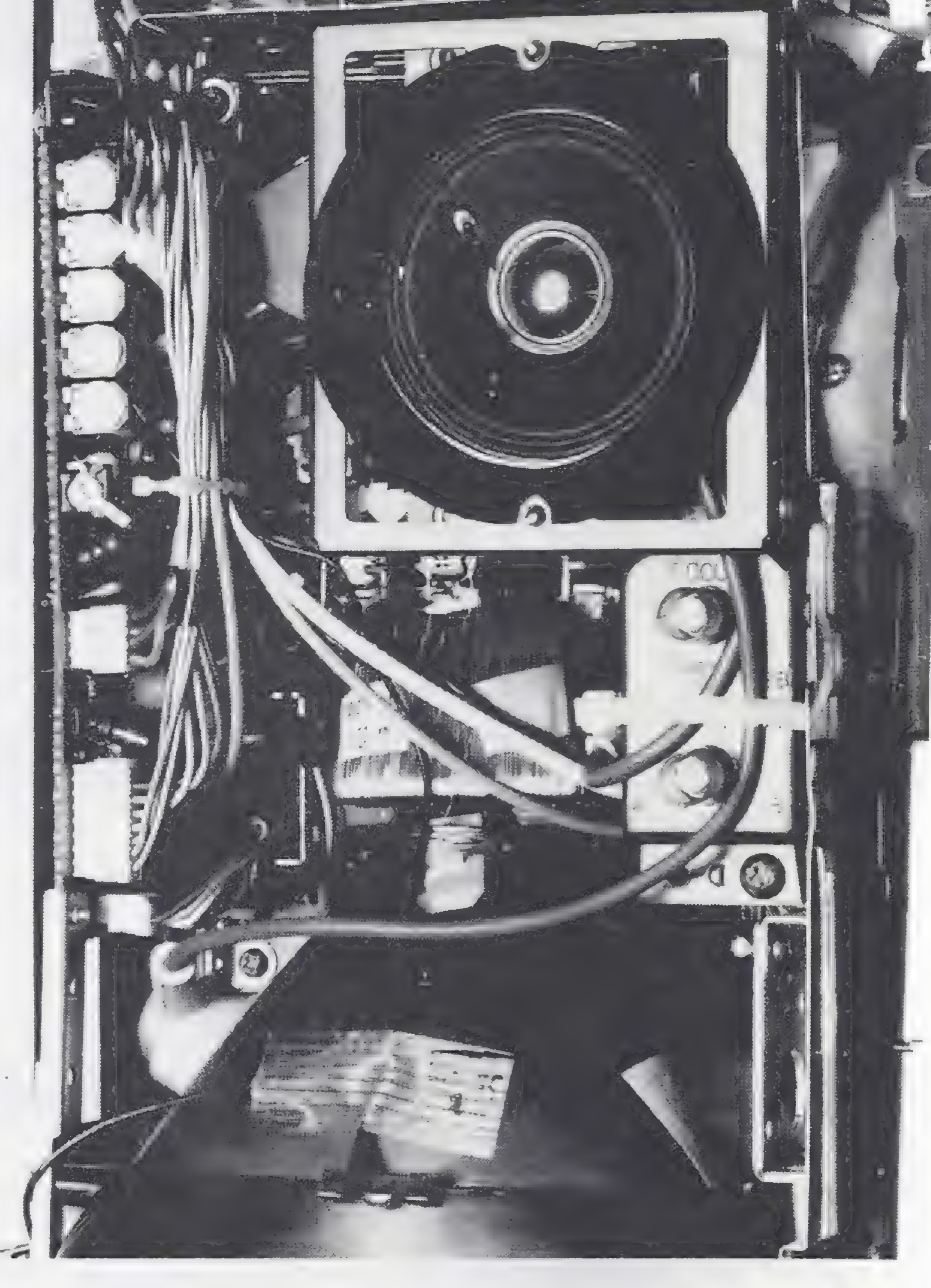
La memoria ROM empleada es de 20 Kbytes, habiendo sido ligeramente alterada la zona que contiene al KERNAL no así el BASIC.

Físicamente el **SX-64** se divide en dos partes; la carcasa principal y la tapa. La segunda es realmente el teclado, que en posición de trabajo se abate y separa totalmente, quedando unido al ordenador por medio de un cable enchufable por ambos extremos.

El teclado está formado por un conjunto de 66 teclas, de ellas 4 son las clásicas de función: F1 a F8, de doble utilidad por su combinación con la tecla SHIFT. Lo compacto y

El ordenador transportable





reducido del teclado le hacen parecer, en un principio, incómodo de manejar. Sin embargo, esta ilusión se desvanece al poco rato, no en vano es una réplica del utilizado en el otro **64**. Probablemente lo que sí se echa en falta es la inclinación de las teclas con respecto a la mesa. SHIFT LOCK dispone de un LED interno que indica su activación. Una vez abatida la tapa-teclado, queda a la vista el panel frontal del ordenador. En el resalta, a la izquierda, el coqueto monitor de 5 pulgadas. Se trata de un monitor de color, cuya resolución y calidad de imagen es excepcional, teniendo en cuenta lo reducido de su tamaño. En el lado opuesto existe una pequeña puerta en sentido vertical. Abriendo el clip que actúa como cerradura aparece una serie de botones de control, destinados a modificar color, contraste, brillo, estabilidad vertical y sonido. Igual que en cualquier monitor comercial. Que estén escondidos tras una tapa hace que al menos no resulten molestos durante la inserción y extracción de los *diskettes*. Existen otros dos controles adicionales, ajustables sólo con destornillador, que llevan las etiqueta de subbrillo y subcontraste. Finalmente se sitúa un pulsador de Reset entre los controles de color y estabilidad vertical.

El monitor puede visualizar línea de hasta 40 caracteres de longitud. El número de línea visualizables simultáneamente es de 25, aunque la pantalla actuará en modo ventana, permitiendo que aparezcan sucesivamente hasta 240 líneas.

En cuanto a capacidades gráficas, este monitor ofrece las mismas características que el modelo clásico: un formato de 200 × 320 puntos y un total de hasta 16 colores. También está observada la posibilidad de utilizar dos conjuntos de caracteres. El normal es el que se utiliza en el modo texto. El alternativo es de gran utilidad para elaborar gráficos con los bloques predefinidos, que van igualmente indicados en la parte frontal de algunas teclas, para su uso directo. En la parte central de panel está situada la unidad de *diskettes*, desti-



nada a alojar discos estándar de 5-1/4 pulgadas. Justo encima de la unidad aparece una ventana, en realidad una caja, de dimensiones idénticas. En el modelo **SX-64** sirve para guardar los *diskettes* que se vayan a necesitar. No obstante puede alojar una segunda unidad de *diskettes,* con lo que el modelo resultante será el **DX-64**.

En realidad, la unidad utilizada es básicamente la clásica **1541**. Es una pena que no se le haya dotado de otro tipo de unidad sensiblemente más rápida, pues es notoria la lentitud del **1541**. La máxima capacidad de información acumulable por *diskette* queda en los discretos 174 Kbytes, una vez formateado.

En la posición de transporte el asa, que pivota a ambos laterales de la carcasa paralelepipédica, queda alineada permitiendo que el traslado sea relativamente confortable. Su peso, aunque no excesivamente incómodo, llega a hacerse molesto si el desplazamiento es prolongado. La parte destinada a la mano es de goma rugosa, para evitar que se escurra con la transpiración.

Durante la utilización del ordenador, el asa se utiliza para disponer el monitor en la forma más cómoda de trabajo para el usuario, pues lleva distintos topes que permiten ir variando el grado de inclinación.

La parte superior de la carcasa muestra una pequeña ventana, cerrada por dos contraventanas, destinada a alojar los cartuchos de ROM enchufables. Su aspecto y la idea de las dos semitapas, que ceden al introducir el cartucho, recuerdan el mecanismo similar utilizado por **Atari** en sus recientes modelos.

La parte posterior del ordenador, en realidad la base cuando se le transporta, es tan interesante como en la mayoría de los ordenadores. En ellas se disponen los feos conectores de conexión del equipo con otras cosas diversas. Gran parte de la superficie ha sido destinada al radiador de disipación del calor, produciú do internamente por ciertos componentes. Las aletas de refrigeración no dejan de conferirle un aspecto relativamente atractivo, desapareciendo

El ordenador transportable

sólo para alojar una chapa en la que van impresas las advertencias y cuidados a observar, así como el modelo e identificaciones de control. La línea superior, situada sobre el radiador, es el lugar elegido para colocar los conectores. De derecha a izquierda, lo primero que vemos es el conmutador de encendido, que da paso a la alimentación. Seguídamente encontramos la tapa, desenroscable, del portafusibles. Un lugar francamente cómodo, cuya facilidad de acceso permite la comprobación del fusible si se produce la nunca deseada avería. Inmediatamente a continuación está el zócalo de tres polos para el conector que trae la alimentación, en corriente alterna, desde la red.

Más o menos en el centro, queda dispuesto el útil *port* de usuario, que aparece como una lengua procedente de la tarjeta de circuito impreso actuando directamente como conector. A su lado se ven otros dos conectores redondos, cuya misión es proporcionar salida para la impresora, uno y señales de audio y vídeo para un monitor externo, el otro. Para concluir encontramos los dos típicos conectores destinados a los *joysticks*.

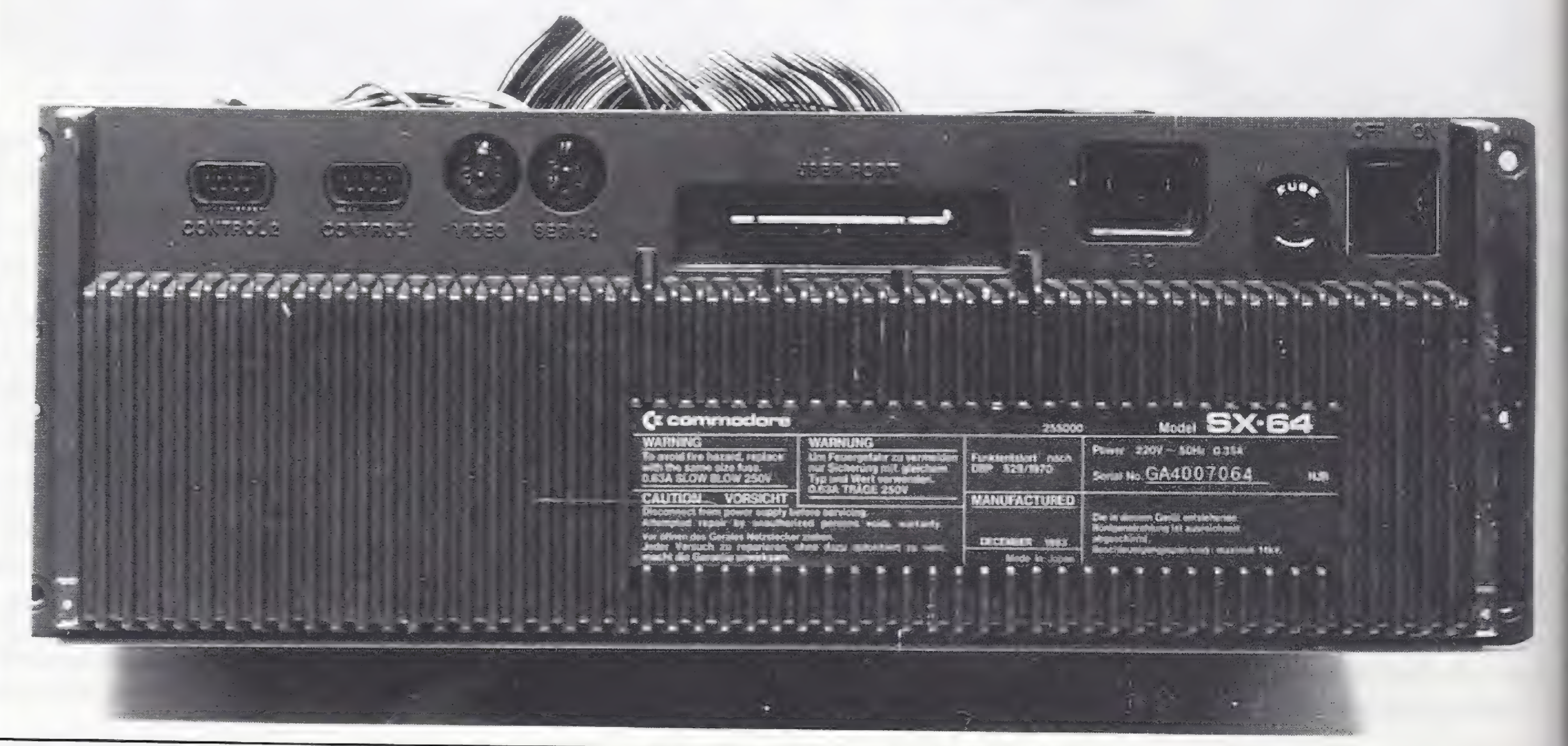
Se echa en falta la conexión destinada a la unidad de *cassettes*, pero es probable que **Commodore**, lo haya omitido en vista de que este modelo dispone de una unidad de *diskettes* como mínimo. Sin embargo, esta carencia limita la utilización directa de programas grabados en *cassette*, a no ser que se haga la transferencia a *diskette* por medio de un 64 clásico; pero, ¿qué ocurre con los programas en cinta que lleven protección anticopia?

Otra carencia que se advierte es la falta de algún tipo de piloto indicador de encendido del aparato.

Una vez que se conecta la alimentación al sistema, el LED indicador de funcionamiento de la unidad de *diskettes* se enciende, denotando que el ordenador busca información contenida en ella. Simultáneamente, el tubo catódico del monitor va caldeándose hasta que la pantalla alcanza su luminosidad normal. Sigue siendo la misma pantalla de colores azul pálido y gris desvaido. Con un monitor de tan reducidas dimensiones resulta que la única forma válida que encontramos de hacer que un texto sea realmente legible es una combinación de colores en la línea de blanco contra gris, como fondo y caracteres en primer plano. De todas maneras, para utilizaciones prolongadas es aconsejable conectar el segundo monitor, preferiblemente monocromático cuando sólo se desea trabajar con modo texto.

El complemento idóneo para que el ejecutivo trabaje con él viene a ser, en un principio, un tratamiento de textos, tal como el potente **Easywriter**, y una hoja de trabajo, en la línea del **Calc Result.** No conviene olvidar el amplio conjunto de paquetes de aplicación desarrollado para el **64**, e incluso la posibilidad de conectar el cartucho **CP/M-80**, para mayor aprovechamiento del numeroso *software* (en inglés) disponible en los mercados.

En conclusión, estamos ante un serio contendiente en la gama de los ordenadores transportables (no confundir estos con los portátiles), que puede ejercer una seria competencia con otros modelos, a pesar de la lentitud de su unidad de disco interna.



La revista imprescindible para todo el usuario de
Ordenadores COMMODORE

commodore Magazine



Aproveche ahora esta irrepetible oportunidad para suscribirse a COMMODORE MAGAZINE. Envíe **HOY MISMO** la tarjeta adjunta, que no necesita sobre ni franqueo. Deposítela en el buzón más cercano. Inmediatamente recibirá su primer ejemplar de COMMODORE MAGAZINE y así durante un año (12 ejemplares).

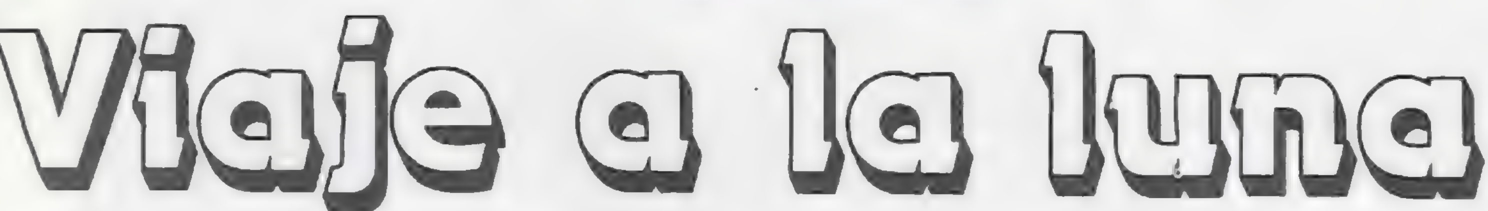
commodore Magazine

Bravo Murillo, 377.
Tel. 733 96 62
Madrid - 20

Concurso







Viaje a la luna

Desde Valladolid a la Luna podemos ir con este programa que, desde dicha ciudad, nos envía Isaac Llopis Ruiz. El juego consiste en llevar hasta la Luna, sano y salvo, un platillo volante, y hacerlo aterrizar, una vez allí, en una base de aterrizaje del subsuelo lunar. El viaje se desarrolla en dos partes y, ni que decir tiene que, a lo largo del camino hay que salvar muchos obstáculos y peligros, que si bien pueden acabar con el viaje en cualquier momento, lo hacen también más entretenido e interesante. La primera parte se trata de aterrizar la nave en una plataforma de aterrizaje subterránea a la que se accede a través de un estrecho túnel, practicado en la roca lunar. En este túnel se pone a prueba la destreza del piloto que debe conducir la nave sin chocar con las paredes.

Para llevar la nave a su destino es necesario utilizar un *joystick*, que se conectará al port 2 del ordenador.

Sólo nos queda esperar que el programa sea de vuestro agrado y desearos un feliz viaje.

```
10 REM***VIAJE A LA LUNA*** LLOPIS
20 POKE53280,14:POKE53281,14:PRINT"VIAJE A LA LUNA"
25 PRINT"────────────────"
30 PRINT"ESTE JUEGO CONSISTE EN LLEGAR A LA"
40 PRINT"LUNA Y APARCAR EN LA BASE LUNAR SIN "
50 PRINT"SER INTERCEPTADO Y SIN CHOCAR CON"
60 PRINT"LAS PAREDES DE LA GRUTA LUNAR"
62 PRINT"DEBERA USAR UN MARDO O JOYSTICK"
65 PRINT"PARA MOVER SU PLATILLO VOLANTE"
70 PRINT"DEBERA ENCHUFARLO EN EL PORT 2"
72 PRINT"UNA VEZ VISTO PULSAR UNA TECLA"
75 GETE$:IFE$=""THEN75
95 VO=54296:WA=54276:H=54273:AT=54277:LO=54272:PH=54275:PL=54274:SU=54278
```

```
100 V=53248
110 POKE53280,9:POKE53281,14:FORT=53285TO53294:READCO:POKET,CO:NEXTT
120 DATA0,13,6,6,6,7,2,2,2,2
130 FORT=12288TO12350:READA:POKET,A:NEXTT
140 FORT=12352TO12414:READA:POKET,A:NEXTT
150 FORT=12416TO12478:READA:POKET,A:NEXTT
160 FORT=12480TO12542:READA:POKET,A:NEXTT
170 FORT=12544TO12606:READA:POKET,A:NEXTT
175 FORT=12608TO12625:POKET,0:NEXT
180 FORT=12626TO12662STEP3:POKET,64:NEXTT
190 FORT=12627TO12663STEP3:POKET,0:NEXTT
200 FORT=12628TO12664STEP3:POKET,1:NEXTT
210 FORT=12665TO12670:READA:POKET,A:NEXTT
220 FORT=12672TO12734:READA:POKET,A:NEXTT
230 FORT=12736TO12798:READA:POKET,A:NEXTT
240 FORT=2040TO2047:POKET,(T-1848):NEXTT
245 U=0:UU=0
247 PRINT"";:IFUU=0THEN268
248 PRINT"                    "
249 PRINT"    XXX                "
250 PRINT"    XXX               "
251 PRINT"                     "
252 PRINT"                    "
253 PRINT"                   "
254 PRINT"                   ",,,"                    ",,,"                "
255 PRINT"              ",,,"                ",,,"       "
256 PRINT"       "
260 POKEV+6,24:POKEV+7,51:T=80:TR=-9:X=249:R=232:H=INT(X/256):L=X-256*H
268 POKEV+21,121:POKE53276,113:POKE53277,(97+8*U):POKE53271,(64+8*U)
269 POKEV+6,24:POKEV+7,51:T=80:TR=-9:X=249:R=232:H=INT(X/256):L=X-256*H
270 POKEV+16,64+H:POKEV+12,10
280 POKE53278,(PEEK(53278)AND0)
285 POKE53278,0
300 FORW=0TO2300
310 IFW/11=INT(W/11)THENK=80+INT(125*RND(1)):TR=-TR:J=100+INT(125*RND(1))
320 Q=20*W-220*INT(W/11):T=T+TR
330 POKEV+1,R:POKEV,L:POKE53278,0:POKEV+13,T
339 POKEV+11,29+Q
340 POKEV+10,K+10:POKEV+8,Q:POKEV+9,J
345 GOSUB10000:GOSUB10000
350 NEXTW
2000 FORT=0TO30:NEXTT:RETURN
2001 DATA0,0,0,233,0,0,40,0,0,170,0,1,85,64,10,170,160,42,170,168,85,85,85
2010 DATA235,174,186,235,174,186,235,174,186,85,85,85,170,170,170,42,170,168
2020 DATA5,85,80,2,170,128,0,170,0,0,20,0,0,0,0,0,0,0,0,0,0
2030 DATA0,0,0,107,0,0,40,0,0,170,0,1,85,64,10,170,160,42,170,168,85,85,85
2040 DATA186,235,174,186,235,174,186,235,174,85,85,85,170,170,170,42,170,168
2050 DATA5,85,80,2,170,128,0,170,0,0,20,0,0,0,0,0,0,0,0,0,0
2060 DATA0,0,0,233,0,0,40,0,0,170,0,1,85,64,10,170,160,42,170,168,85,85,85
2070 DATA174,186,235,174,186,235,174,186,235,85,85,85,170,170,170,42,170,168
2080 DATA5,85,80,2,170,128,0,170,0,0,20,0,0,0,0,0,0,0,0,0,0
2090 DATA0,0,0,126,0,1,255,128,7,255,224,12,30,48,31,255,248,60,63,28
2100 DATA61,182,220,127,247,254,127,247,254,127,243,254,127,251,254,127,255,254
2110 DATA63,255,252,63,255,60,31,191,120,15,128,112,7,255,224,1,255,128,0,126,0
2120 DATA0,0,0,85,85,86,21,85,84,5,85,0,1,85,0,1,85,0,0,85,0,0,85,64,0,20,16
2130 DATA0,20,4,0,20,4,0,20,8,0,20,4,0,20,4,0,20,16,0,85,64,0,85,0,1,85,0
2140 DATA1,85,0,5,85,0,21,85,84,85,85,86,128,0,2,128,0,2
2150 DATA10,170,160,10,105,160,2,105,128,2,170,128,128,170,2,160,170,10
2160 DATA42,190,168,10,190,160,2,255,128,2,255,128,10,190,160,42,190,168
```




```
2170 DATA160,190,10,128,190,2,2,170,128,10,170,160,42,170,168,40,130,40
2180 DATA32,130,8,32,130,8,0,130,0
2190 DATA8,0,0,8,0,0,8,0,0,8,0,0,8,0,0,28,0,0,62,0,0,54,0,0,127,0,0,119,0,0
2200 DATA127,0,0,119,0,0,127,0,0,119,0,0,127,0,0,119,0,0,127,0,0,62,0,0
2210 DATA127,0,0,255,128,0,221,128,0,221,128,0
3000 POKEV+21,0:PRINT"▀▀▀▀▀YA ESTA PROXIMO A LA LUNA. AHORA "
3010 PRINT"▀▀DEBERA INTRODUCIR SU PLATILLO HASTA"
3020 PRINT"▀▀LA PLATAFORMA NEGRA SIN TROPEZAR":FORY=0TO2500:NEXTY:PRINT"▀"
3040 POKEV+21,135:POKE53279,((PEEK(53279))AND0):POKE53280,6:POKE53281,14
3050 FORT=0TO15:POKEV+T,0:NEXTT
3060 X=265:R=60:H=INT(X/256):L=X-256*H:POKE53276,7:POKE53277,7:POKE53271,128
3070 POKEV+16,128+7*H:POKE53279,0:POKEV,L:POKEV+1,R
3080 PRINT"                    ▀*"
3081 PRINT"▀  *                  ▀":PRINT
3082 PRINT"         *         ▀"
3083 PRINT"         *                  *":PRINT
3084 PRINT"                                *  ":PRINT
3085 PRINT"        *    *                   ":PRINT
3086 PRINT"   *                     *   ":PRINT:O=5:GOSUB6000
3087 PRINT"             *             ":PRINT:O=7:GOSUB6000
3088 PRINT"                                *  ":PRINT:O=35:GOSUB6000
3089 PRINT"        *    *                   ":PRINT:O=40:GOSUB6000
3090 PRINT"   *                     *   ":PRINT:O=70:GOSUB6000
3092 PRINT"                    ▀*":GOSUB6000
3093 PRINT"▀  *                  ▀":PRINT:GOSUB6000
3094 PRINT"         *         ▀":GOSUB6000
3095 PRINT"         *                  *":PRINT:GOSUB6000
3096 PRINT:GOSUB6000:PRINT:GOSUB6000:PRINT:GOSUB6000:O=90
3097 PRINT:GOSUB6000:PRINT:GOSUB6000:PRINT:GOSUB6000:PRINT:GOSUB6000:O=150
3100 PRINT"                      ▘   ▀     ◥ ▀":GOSUB6000:O=160
3110 PRINT"▀                     ◥     ▀▀   ▀ ◥▀":GOSUB6000:O=170
3120 PRINT"▘ ▀                   ◥           ▀  ◥    ":GOSUB6000:O=180
3130 PRINT"▀▘   ▌          ▘                   ▌  ▘    ":GOSUB6000:O=190
3140 PRINT"▀▘   ▌          ▘                   ▌  ▘    ":GOSUB6000:O=200
3150 PRINT"▀▘  ▌          ▘                    ▌  ▘    ":GOSUB6000
3160 PRINT"▀▘  ▌          ▘                    ▌  ▘    ":GOSUB6000
3170 PRINT"▀▘ ▌           ▘      ▌    ▘  ▌  ▘     ▌  ▘    ":O=220:GOSUB6000
3180 PRINT"▀▘   ▌         ▘     ▌       ▘     ▌  ▘    ":O=240:GOSUB6000
3190 PRINT"▀▘ ▌            ▘    ▌         ▘   ▌  ▘    ":O=260:GOSUB6000
3200 PRINT"▀▘ ▌            ▘   ▀▀▘ ▌            ▘ ▌  ▘    ":O=280:GOSUB6000
3210 PRINT"▀▘ ▀            ▘  ▌               ▘ ▌  ▘    ":O=300:GOSUB6000
3220 PRINT"▀▘     ▌        ▘  ▌               ▘ ▌  ▘    ":GOSUB6000
3230 PRINT"▀▘     ▌        ▘  ▌          ▘ ▌         ▘   ":GOSUB6000
3240 PRINT"▀▘    ▌               ▘  ▌          ◥▘   ":O=350:GOSUB6000
3250 PRINT"▀▘   ▌               ▘   ▌           ▘   ":O=375:GOSUB6000
3260 PRINT"▀▘   ▌              ▘    ▌      ▘  ▌     ▘ ":O=400:GOSUB6000
3270 PRINT"▘                     ▌     ▌    ▀  ▌  ▀   ▗"
3280 POKE1983,224:POKE56255,7
3281 POKE2023,224:POKE56295,7
3290 POKEV+14,45:POKEV+15,199
3291 VO=54296:WA=54276:H=54273:AT=54277:LO=54272:PH=54275:PL=54274:SU=54278
3292 POKEVO,15:POKEAT,32:POKESU,255:POKEAT+7,32:POKESU+7,255:POKEH,17:POKELO,3
3293 POKEWA,17:POKEWA+7,17:POKEH+7,16:POKELO+7,3:POKEPH,8:POKEPL,8
3300 FORW=0TO100000
3310 POKEV,L:POKEV+1,R:POKEV+5,0:POKEV+4,0:GOSUB20000
3320 POKEV+2,L:POKEV+3,R:POKEV+1,0:POKEV,0:GOSUB20000
```

```
3340 POKEV+4,L:POKEV+5,R:POKEV+3,0:POKEV+2,0:GOSUB20000
3350 NEXTW
3940 POKEV+21,135:POKE53279,((PEEK(53279))AND0):POKE53280,6:POKE53281,14
3960 X=265:R=60:H=INT(X/256):L=X-256*H:POKE53276,7:POKE53277,7:POKE53271,128
3970 POKEV+16,128+7*H:POKE53279,0:POKEV,L:POKEV+1,R
4000 GOTO4000
6000 FORT=0TOO:NEXTT:RETURN
7000 POKEV+16,128
7010 POKEWA,129:POKEWA+7,0:POKEWA+14,0:POKEVO,15
7020 POKEWA,17:POKEWA+7,17:POKEH,32:POKEH+7,33:POKELO,94:POKELO+7,75
7100 FORW=199TO58STEP-1:POKEV+15,W:IFW<75THENGOSUB2000
7101 IFW>180THENGOSUB2000
7110 NEXTW
7120 FORF=45TO0STEP-1
7121 IFF>15THENGOSUB2000
7125 POKEV+14,F:NEXTF
7130 POKEV+16,0
7131 FORE=255TO58STEP-1
7132 POKEV+14,E:IFE<72THENGOSUB2000
7133 NEXTE
7140 PRINT"■SRQ▶▶▶▶▶▶VIVA LA PEPA"
7200 FORT=54272TO54290:POKET,0:NEXTT
```




```
8000 GOTO8000
10000 P=(15AND(PEEK(56320)))
10010 P1=PAND1
10020 P2=PAND2
10030 P3=PAND4
10040 P4=PAND8
10050 R=R-2.4*P2+4.8*P1
10055 X=X-1.3*P4+2.6*P3
10060 IFR>232THENR=232
10061 IFR<50THENR=50
10062 IFX>320THENX=320
10063 IFX<24THENX=24
10066 H=INT(X/256):L=X-256*H:POKEV+16,64+H
10067 IF(PEEK(53278)AND1)=1THEN10074
10070 RETURN
10074 PRINT"■";(PEEK(53278))
10075 IF(PEEK(53278)AND8)=8THENFORE=8TO11:POKEV+E,0:NEXTE:GOTO10100
10080 FORQ=0TO10:POKEVO,15:POKEAT,0:POKESU,128:POKE54276,129:POKEH,40:POKELO,94
10081 POKE53276,113:GOSUB2000:POKE53276,112:GOSUB2000:POKEVO,0:POKE54276,0:NEXTQ
10090 POKEV+21,0:PRINT"■■■■■■■■■LO SIENTO,FUE INTERCEPTADO, PARA"
10091 PRINT"■■■■■INTENTARLO OTRA VEZ PULSE LA BARRA"
10092 GETE$:IFE$=""THEN10092
10093 IFE$=" "THENRUN
10095 END
10100 IFU=0THENU=1:GOTO10300
10110 IFUU=0THENUU=1:GOTO10305
10120 GOTO3000
10300 POKEV+21,0:PRINT"■■■■■■■■■■■■■■■YA HA RECORRIDO 130000KM.":GOTO10310
10305 POKEV+21,0:PRINT"■■■■■■■■■■■■■■■YA HA RECORRIDO 260000KM."
10310 PRINT"■■■■■■■■■■■PREPARADO PARA CONTINUAR":FORY=0TO1500:NEXTY:GOTO247
20000 P=(15AND(PEEK(56320)))
20010 P1=PAND1
20020 P2=PAND2
20030 P3=PAND4
20040 P4=PAND8
20050 R=R-.5*P2+1*P1
20055 X=X-.25*P4+.5*P3
20066 H=INT(X/256):L=X-256*H:POKEV+16,128+7*H
20067 IFL>220ANDL<237ANDR>224THEN7000
20069 IF(PEEK(53279)AND1)=1THEN20180
20070 RETURN
20180 FORQ=0TO10:POKEVO,15:POKEAT,0:POKESU,128:POKE54276,129:POKEH,40:POKELO,94
20181 POKE53276,113:GOSUB2000:POKE53276,112:GOSUB2000:POKEVO,0:POKE54276,0:NEXTQ
20185 GOTO10090
```







Biorritmos

Desde Madrid, Gines Plaza López nos envía este programa, BIORRITMOS, para el **Commodore 64.** Los biorritmos, como mucha gente sabe, son ciclos biológicos que parten de cero el día de nuestro nacimiento, y que van evolucionando a lo largo de nuestra vida, pasando por máximos y mínimos y, según dicen algunos, determinando en cierto modo nuestra suerte, nuestro potencial y nuestro destino. Hay tres ciclos o biorritmos fundamentales, con distinto período de repetición cada uno de ellos que son:

El biorritmo físico que determina nuestro potencial físico y que se repite con una periodicidad de 23 días.

El biorritmo emocional, determinante de nuestro estado afectivo y con un período de 28 días.

Y por último, el biorritmo intelectual, que marca nuestras capacidades intelectuales y de trabajo creativo y que se repite cada 33 días.

Con este programa podremos conocer en cualquier momento el estado y la evolución de cada uno de estos ciclos.

El programa comienza pidiendo varios datos como son el nombre y la fecha de nacimiento de la persona que desea ver su biorritmo. Luego pide la fecha de inicio del biorritmo y cuantos días se desean ver a partir de dicha fecha. Los datos de las fechas hay que escribirlos separados por comas, por ejemplo una fecha válida es 3, 4, 1984.

Después de introducidos todos los datos, el programa realiza sus cálculos, nos dice cuantos días han transcurrido desde la fecha de nacimiento a la fecha de inicio del biorritmo y nos presenta un menú para que escojamos cual de los 3 biorritmos deseamos ver en pantalla.

El biorritmo escogido se presenta en forma gráfica en la pantalla, correspondiendo el eje vertical a los días y el eje horizontal al valor del biorritmo para cada día. Los máximos aparecen a la derecha de la pantalla y significan buenas perspectivas, por el contrario los mínimos aparecen a la izquierda y no presagian nada bueno. Por último, el programa nos pregunta si deseamos ver algún otro biorritmo, terminando en el caso de que la respuesta sea negativa.

A partir de ahora, y con este programa, antes de salir a la calle habrá que consultar, no sólo el horóscopo, sino también el estado de los biorritmos.

```
5 PRINT"▯"
10 DIM MESES$(12)
15 PRINT"▩                B I O R R I T M O S          "
20 INPUT"▩SU NOMBRE, POR FAVOR";NOMBRE$
30 PRINT"▩▩ENTRADA DE FECHAS: (DD,MM,AAAA)"
50 INPUT"▩▩▩FECHA DE NACIMIENTO";DI,MI,AI
60 INPUT"▩▩FECHA DE INICIO";DF,MF,AF
100 INPUT"▩▩▩PARA CUANTOS DIAS LO QUIERE";B
120 FOR I=1 TO 12:READ MESES$(I):NEXT
130 A=AI:M=MI:D=DI:GOSUB 1000:ND(0)=ND
140 A=AF:M=MF:D=DF:GOSUB 1000:ND(1)=ND
150 D=ND(1)-ND(0)
170 DEF FNA(X)=(X-1900)/4
180 D=D+INT(FNA(AF))-INT(FNA(AI))
210 CF=2*π/23
220 CE=2*π/28
```

Concurso

```
230 CI=2*π/33
240 PRINT"□■■■■■■■■■BIORRITMO DE ";NOMBRE$
250 PRINT"■A PARTIR DEL";DF;"DE ";          MESES$(MF);" DE";AF
260 PRINT"■■LOS DIAS TRANSCURRIDOS DESDE LA FECHA"
270 PRINT"■DE NACIMIENTO HASTA EL COMIENZO"
280 PRINT"■DEL BIORRITMO SON ■";D
290 PRINT"■■QUE BIORRITMO DESEA?"
300 PRINT"■ <F>: FISICO"
310 PRINT"■ <E>: EMOCIONAL"
320 PRINT"■ <I>: INTELECTUAL"
330 GET A$:IF A$="" THEN 330
340 IF A$="F" THEN 400
350 IF A$="E" THEN 500
360 IF A$="I" THEN 600
370 GOTO 330
380 :
400 COEFICIENTE=CF
410 T$="FISICO    ":P$="○"
420 GOSUB 700
425 :
430 PRINT"■DESEA OTRO BIORRITMO? (S/N)"
440 GET D$:IF D$="" THEN 440
450 IF D$="S" THEN 240
460 IF D$<>"N" THEN 440
470 GOTO 900
480 :
500 COEFICIENTE=CE
510 T$="EMOCIONAL    ":P$="●"
520 GOTO 420
530 :
600 COEFICIENTE=CI
610 T$="INTELECTUAL    ":P$="♦"
620 GOTO 420
630 :
700 PRINT"□      ■  BIORRITMO ";T$
710 FOR I=0 TO B-1
720 X=INT(17*SIN((I+D)*COEFICIENTE)+17)
730 IF I<10 THEN PRINT" ";
740 PRINT I;"|";
750 FOR J=1 TO X-1
760 PRINT".";
770 NEXT J
780 PRINT P$
790 NEXT I
800 RETURN
810 :
900 END
910 :
920 REM--------- NOMBRES DE MESES--------
930 :
960 DATAENERO,FEBRERO,MARZO,ABRIL,MAYO
970 DATAJUNIO,JULIO,AGOSTO,SEPTIEMBRE
980 DATAOCTUBRE,NOVIEMBRE,DICIEMBRE
990 REM-----CALCULO DEL DIAS------------
1000 A=A-1:IF M>2 THEN A=A+1
1010 M=M+13:IF M>15 THEN M=M-12
1020 ND=INT(365.25*A)+INT(30.6001*M)+D+1720982
1030 RETURN
```

PROGRAMA commodore COMPROBADO

Concurso

Cruz

CRUZ es un juego para el **Commodore 64,** enviado por Luis Meléndez Aganzos desde Córdoba. El programa comienza dando las instrucciones de juego para, a continuación, borrar la pantalla y presentar en ella el tablero de juego. Este consiste en una matriz de 7 × 7, con las filas numeradas de una a siete y las columnas identificadas por letras, de la A a la F, sobre el que aparece dibujada una figura en forma de cruz. La cruz se supone que está constituida por fichas sobre un tablero, y el objeto del juego es ir comiendo fichas, una cada por cada jugada, hasta que sólo quede una. Las reglas para comer fichas son las siguientes: Para comerse una ficha hay que saltar sobre ella, con otra ficha, hasta un espacio vacío, igual que en el juego de las damas. Sólo se puede saltar en sentido vertical u horizontal, pero no en diagonal y sólo se puede comer una ficha por jugada. La jugada se introduce por el teclado, respondiendo a las preguntas ¿MUEVE DESDE?, y ¿HASTA? Después de introducida la jugada, el programa se encarga de comprobar si es válida o no, y sólo la admite si la jugada es correcta. El juego continúa de esta manera hasta que el jugador alcanza su objetivo o bien hasta que decide rendirse, lo que hace pulsando la tecla cero.

```
4 PRINT"":PRINT"ESCRIBA PRIMERO LA LETRA"
5 PRINT"Y LUEGO EL NUMERO"
6 PRINT"PARA RENDIRSE, PULSE "
7 PRINT"PULSE UNA TECLA PARA EMPEZAR"
8 GET Z$:IF Z$="" THEN 8
10 PRINT"":PRINT""
20 POKE 53280,7:POKE 53281,2
30 PRINTTAB(12);"A B C D E F G"
35 PRINTTAB(12);"─────────────"
40 FOR I=1 TO 7
50 PRINTTAB(8);I;"|":PRINTTAB(11);"|"
60 NEXT I
80 PRINT""
100 FOR I=1 TO 7:PRINT
110 IF I=4 THEN PRINTTAB(12);"● ● ●   ● ● ●":GOTO 140
120 IF I=3 OR I=5 THEN PRINTTAB(12);"● ● ● ● ● ● ●":GOTO 140
130 PRINTTAB(12);". . ● ● ● . ."
140 NEXT I
150 PRINT""
160 INPUT"MUEVE DESDE:";A$
170 IF A$="0" THEN 510
180 IF LEN(A$)<>2 THEN PRINT"":GOTO160
190 INPUT"HASTA:";B$
195 IF B$="0" THEN 510
200 IF LEN(B$)<>2 THEN PRINT"":GOTO 190
205 PRINTTAB(14);"      "
207 PRINTTAB(8);"      "
215 A1=ASC(LEFT$(A$,1)):A2=VAL(RIGHT$(A$,1))
220 B1=ASC(LEFT$(B$,1)):B2=VAL(RIGHT$(B$,1))
230 IF A2>7 OR A2<1 THEN PRINT"":GOTO160
240 IF B2>7 OR B2<1 THEN PRINT"":GOTO190
250 IF A1>71 OR A1<65 THENPRINT"":GOTO160
260 IF B1>71 OR B1<65 THENPRINT"":GOTO190
280 IFA1<>B1 AND A2<>B2 THEN PRINT"":GOTO160
290 IF ABS(A2-B2)<>2 AND ABS(A1-B1)<>2 THEN PRINT"":GOTO 160
300 M1=1114+2*(A1-64)+80*A2
302 M2=1114+2*(B1-64)+80*B2
304 M3=1114+2*((A1+B1-128)/2)+80*((A2+B2)/2)
320 IF PEEK(M1)<>81 THEN 470
330 IF PEEK(M2)<>32 THEN 470
340 IF PEEK(M3)<>81 THEN 470
400 POKE M1,32
410 POKE M2,81
420 POKE M2+54272,7
430 POKE M3,32
450 CT=CT+1
460 IF CT>30 THEN 500
470 PRINT"":GOTO 160
480 PRINT"
```





```
490 PRINT"                    ]]]"
500 PRINTTAB(5);"  E N H O R A B U E N A  "
510 PRINTTAB(7);"  DESEA OTRA PARTIDA ?"
520 GET A$
530 IF A$="S" THEN RUN
540 IF A$="N" THEN END
550 GOTO 520
```

PROGRAMA commodore PUBLICADO

Cartas

Daniel Cocho González, lector de **Commodore Magazine** en Madrid, nos remite un interesante programa que él ha llamado DIBUJO JOY, nombre muy adecuado para este programa, ya que permite realizar dibujos con el *joystick*, sobre la pantalla de alta resolución del **64.** El programa comienza presentando en pantalla las instrucciones para la realización de los dibujos. Estas instrucciones son sencillas; para dibujar una línea basta mover el *joystick* en la dirección deseada, que puede ser horizontal, vertical o diagonal. En cualquier momento, se puede pulsar el botón de disparo del *joystick* lo que equivale a "levantar el lápiz del papel", de esta forma el punto deja de imprimir, aunque puede seguir moviéndose por la pantalla.

```
1 PRINT"":PRINTCHR$(147):PRINTTAB(8)"                      003"
2 PRINTTAB(9)"  ABEZA DE SERIE 003"
3 PRINTTAB(10)"      IRO  OFT 1984 ":POKE650,128
4 FORI=1TO2000:NEXT:PRINTCHR$(147)
5 DIMZ$(4,1000):DIMY(4):PRINT"":POKE53280,2:POKE53281,2:GOTO50
6 DIMZ$(4,1000):DIMY(4):INPUT"   UMERO DE PAGINAS DEL TEXTO (MAX.4):";N
7 IFN<1ORN>4THENPRINT" UMERO DE PAGINAS NO RECONOCIDO"
8 IFN<1ORN>4THENGOTO6
9 PRINTCHR$(147);TAB(8)"  EFINICION DE PAGINAS "
10 PRINT"  AX. NUMERO DE CARCTERES:1000 POR PANTA LLA.";
11 PRINT" UANDO LA PANTALLA SE BORRE ESCRIBA EL TEXTO."
12 PRINT:PRINT:PRINTTAB(5)"( ARA SEGUIR PULSE UNA TECLA)"
13 GETWA$:IFWA$=""THEN13
14 FORI=1TON:PRINT"":PRINTTAB(12)"      ";I:PRINT" UMERO APROXIMADO DE ";
15 PRINT"CARACTERES DE LA PAGINA";I:INPUTY(I):PRINT"":POKE53280,12:FORR=1TOY(I)
16 GETZ$(I,R):IFZ$(I,R)=""THEN16
17 PRINTZ$(I,R);:NEXTR:NEXTI:POKE53280,2:PRINT"":GOTO50
18 PRINT"":FORI=1TON:FORR=1TOY(I):PRINTZ$(I,R);:NEXTR
19 GETQA$:IFQA$=""THEN19
20 PRINT"":NEXTI:GOTO50
21 PRINTCHR$(147):F$="":PRINT:INPUT" ITULO DEL TEXTO EN  ATASSETTE";F$
22 IFLEN(F$)=0THEN21
30 OPEN1,1,1,F$
31 PRINT#1,N:FORI=1TON:PRINT#1,Y(I):FORR=1TOY(I):PRINT#1,Z$(I,R);:NEXTR:NEXTI
32 CLOSE1:GOTO50
40 DIMZ$(5,1000):DIMY(5):PRINT"":INPUT"TITULO ";F$:OPEN1,1,0,F$:INPUT#1,N
41 FORI=1TON:INPUT#1,Y(I):FORR=1TOY(I):GET#1,Z$(I,R):NEXTR:NEXTI:CLOSE1:GOTO18
50 PRINT"":PRINTTAB(14)"                  L":PRINT
51 PRINTTAB(10)" (L)  LEER CINTA       "
52 PRINTTAB(10)" ( )   SCRIBIR TEXTO   "
53 PRINTTAB(10)" (-)   OMPROBAR TEXTO "
54 PRINTTAB(10)" ( )   ALVAR TEXTO     "
55 PRINTTAB(10)" (_)   IN DE PROGRAMA "
56 PRINT:PRINT:PRINT:PRINT:PRINTTAB(16)" PCION";:INPUTD$
57 IFD$="L"THENRUN40
58 IFD$=" "THENRUN6
59 IFD$="-"THEN18
60 IFD$=" "THEN21
61 IFD$="_"THENEND
62 PRINT:PRINT" PCION NO RECONOCIDA":FORT=1TO200:NEXT:GOTO50
```

PREMIADO CON 5.000 PESETAS

Concurso

Crucero estelar

De nuevo una misión de combate en las estrellas. En este caso se trata de proteger un conboy de naves, de los ataques de cazas enemigos. El programa comienza presentando un título, luego se encarga de cargar los datas de los diversos sprites, datas que hay que copiar con sumo cuidado, para pasar a preguntar, a continuación, si se desean instrucciones o no. A continuación comienza el programa principal, creando sobre la pantalla una simulación muy interesante de la cabina de una nave de combate. En esta cabina aparece un radar para indicar la posición de las naves enemigas aún cuando no estén al alcance visual, además este radar lleva dos marcas a los lados para indicar el momento en el que las naves enemigas entran dentro del radio de acción del laser. Aparecen también en la cabina dos indicadores, uno de ellos mide el combustible de que se dispone, mientras que el otro es un marcador de puntuación que se incrementa cada vez que una nave enemiga es alcanzada. El último elemento esencial de la cabina es el cristal delantero, por donde se verá aparecer a las naves enemigas cuando estén dentro del radio visual, y que lleva en su centro una mira para el laser, sobre la que hay que situar al enemigo para alcanzarle. Las naves enemigas son resistentes y por ello, solo después de cinco impactos hacen explosión. Por cada nave destruida aparecerá una marca en el ángulo inferior izquierdo de la pantalla.

```
10 REM *********************
12 REM *                   *
13 REM *  CRUCERO ESTELAR  *
14 REM *                   *
15 REM *********************
16 REM
17 REM
20 POKE53280,12:POKE53281,12:GOSUB2500
50 REM ** REGISTRO DE SONIDO **
51 SD=54272
60 DIMSC$(5),SC(5),SH(5):FORJ=1TO5:SC$(J)="**********":SC(J)=0:SH(J)=0:NEXTJ
100 REM ** INICIALIZA REGISTROS DE SPRITES
101 V=53248:POKEV+40,6:POKEV+44,6:POKEV+45,6:POKEV+46,2:POKE53275,255
105 POKE2045,244:POKE2046,245:POKE2047,246
120 REM ** DECLARACION DE VARIABLES **
125 G=0:HI=0
130 SC=0:X=0:Y=0:I=0:SI=0:XP=0:XE=0:XR=0:XD=0:N=0:JY=0:M=0:Q=0:S=0:TI$="000000"
135 YR=0:CL=0:F=500:A=0:SH=0
140 IF G=1THENGOTO190
150 IFG=0THENGOSUB5000
160 REM * OPCION DE INSTRUCCIONES **
161 IFG=0THENPRINT"[illegible] QUIERES INSTRUCCIONES ? (S/N)"
167 GETA$:IFA$=""THEN167
169 IFA$<>"S"ANDA$<>"N"THEN167
171 IFA$="S"THENGOSUB3000
175 G=1:POKE53280,11:POKE53281,12
190 GOSUB1000:GOTO700
200 REM ** MOVIMIENTO IZQUIERDA **
201 POKESD+4,19
203 X=X-6:IFX=<0ANDSI=2THENX=X+255:POKEV+16,0:POKEV+2,X:SI=0
204 IFX=<0ANDSI=0THENX=0
205 IFXP=<0THENXP=0
207 POKEXD+(YR-1)*40,160:POKEXD+54272+(YR-1)*40,5
211 XE=XP:XP=XP-6:IFXP<0THENXP=0
212 XR=1558+INT(XE*12/347)
218 POKEXR+YR*40,160:POKEXR+54272+YR*40,5
220 XD=1558+INT(XP*12/347)
225 POKEXD+YR*40,87:POKEXD+54272+YR*40,6
230 RETURN
```

Cuando se acabe el combustible, un sonido de sirena indicará el final de la misión. Entonces y según la puntuación obtenida, podrá escribirse el nombre del piloto que figurará en una tabla de los 10 mejores. El control de la nave se realiza mediante un *joystick*, conectado al *port* 2 del **64**, que también sirve para disparar el laser. Sin embargo, para aquellos lectores que todavía no dispongan de un *joystick* les bastará con sustituir las líneas 780 a 805 por las siguientes, que permiten control desde el teclado:



```
240 REM ** MOVIMIENTO DERECHA **
241 POKESD+4,19
243 X=X+6:IFX=>256 AND SI=0THENX=X-256:POKEV+16,2:POKEV+2,X:SI=2
244 IFX>90 AND SI=2 THENX=90
248 POKEXD+(YR-1)*40,160:POKEXD+54272+(YR-1)*40,5
250 XE=XP:XP=XP+6:IFXP>347THENXP=347
252 XR=1558+INT(XE*12/347)
254 POKEXR+YR*40,160:POKEXR+54272+YR*40,5
255 XD=1558+INT(XP*12/347)
258 POKEXD+YR*40,87:POKEXD+54272+YR*40,6
260 RETURN
270 REM ** MOVIMIENTO ARRIBA **
271 POKESD+4,19
275 Y=Y-6:IFY<0THENY=0
276 RETURN
280 REM ** MOVIMIENTO ABAJO **
281 POKESD+4,19
285 Y=Y+6:IFY>150 THEN Y=150
290 RETURN
295 REM ** RUTINA DE PUNTUACION **
300 IF YR>2THENSC=SC+10:A=A+1
305 IFA=5THENSC=SC+100
310 PRINT"[control characters]";SC
315 RETURN
320 REM ** IMPRIME FUEL **
325 POKESD+4,18
327 F=INT(F-CL/50)
330 PRINT"[control characters]      [control characters]";F
335 RETURN
340 REM ** RUTINA DE EXPLOSION **
345 POKEV+21,1:POKEXD+YR*40,160:POKEXD+54272+YR*40,5
350 FOR I=0TO7:POKEXD+I*40,160:POKEXD+54272+I*40,5:NEXT
355 POKE1985+SH,87:POKE56257+SH,2:SH=SH+1
357 POKESD+1,6:POKESD+24,15:POKESD+4,129
360 POKEV+10,176:POKEV+11,79:POKEV+21,33:FORI=6TO15STEP.3:POKESD+24,I:NEXT
```

Concurso

```
365 POKEV+14,176:POKEV+15,79:POKEV+21,129:FOR I=0TO40:NEXT
370 POKEV+12,176:POKEV+13,79:POKEV+21,193:FORI=0TO40:NEXT
375 POKEV+14,164:POKEV+15,69:POKEV+29,128:POKEV+23,128
378 FOR I=15TO0STEP-.15:POKESD+24,I:NEXT
379 FOR I=0TO5:NEXT
380 POKESD+24,0:POKEV+21,1:POKEV+29,0:POKEV+23,0:A=0
400 RETURN
500 REM ** RUTINA LASER **
505 PRINT"SQQQQQQQQQ"
507 POKESD+4,129:POKESD+5,15:POKESD+1,40:POKESD,200
510 PRINT"S]]]]]]]]]]]]]]]L╱╱]]]]]]]]]]┐╲╲"
512 POKESD+24,2
520 PRINT"TT]]]]]]]]]]]]]]]]L╱╱]]]]]]]]┐╲╲"
522 POKESD+24,5
530 PRINT"T]]]]]]]]]]]]]]]]]]L╱╱]]]]]]┐╲╲"
532 POKESD+24,9
540 PRINT"T]]]]]]]]]]]]]]]]]]]]L╱╱]]]┐╲╲"
542 POKESD+24,13
550 PRINT"T]]]]]]]]]]]]]]]]]]]]]]L╱╱]┐╲╲"
552 POKESD+24,15
560 PRINT"T]]]]]]]]]]]]]]]]]]]]]]]]L╱╳╲"
600 PRINT"SQQQQQQQQQ"
610 PRINT"]]]]]]]]]]]]]]]   ]]]]]]]]]]   "
612 POKESD+24,15
620 PRINT"TT]]]]]]]]]]]]]]]]   ]]]]]]]]   "
622 POKESD+24,13
630 PRINT"T]]]]]]]]]]]]]]]]]]   ]]]]]]   "
632 POKESD+24,9
640 PRINT"T]]]]]]]]]]]]]]]]]]]   ]]]   "
642 POKESD+24,5
650 PRINT"T]]]]]]]]]]]]]]]]]]]]   ]   "
652 POKESD+24,0
660 PRINT"T]]]]]]]]]]]]]]]]]]]]]      "
665 IFX>173ANDX<185ANDY>75ANDY<83THENGOSUB295
699 RETURN
700 REM ** INICIALIZACION **
705 F=500:A=0
710 YR=0:TI$="000000":N=0
715 X=INT(348*RND(1)):XE=X:XP=X
720 IFX=<255THENSI=0
725 IFX>255THENX=X-256:SI=2
730 Y=INT(150*RND(1))
735 XR=1558+INT(XE*12/347)
745 POKEXR,160:POKEXR+54272,5
750 XD=1558+INT(XP*12/347)
760 POKEXD,87:POKEXD+54272,6
770 REM ** BUCLE PRINCIPAL **
775 POKE2041,241+N:POKEV+2,X:POKEV+3,Y
776 POKEV+21,3:POKEV+16,SI
780 JY=NOTPEEK(56320)AND15
785 IFJYAND1THENGOSUB270
790 IFJYAND2THENGOSUB280
795 IFJYAND4THENGOSUB240
800 IFJYAND8THENGOSUB200
805 IFNOTPEEK(56320)AND16THENGOSUB500
810 IFA=5THENGOSUB340:GOTO710
```

```
811 POKESD+1,100:POKESD+5,219:POKESD+15,28:POKESD+24,15:POKES+4,19
815 CL=VAL(RIGHT$(TI$,2))
820 GOSUB320:IFF<1THENGOTO1500
825 IFCL/4=INT(CL/4)THENN=INT(CL/4):YR=N
830 IFN>3THENN=3
835 IF YR<8THENGOTO850
840 POKEV+21,1:POKEXD+280,160:POKEXD+54272+280,5
845 FORI=1TO2000:NEXT:A=0:GOTO710
850 REM ** MOVIMIENTO ALEATORIO **
851 M=INT(RND(1)*4)+1
855 ON M GOSUB 200,240,270,280
900 GOTO770
1000 REM ** DIBUJO DE PANTALLA **
1005 PRINT"           ▀▀"
1010 PRINT"                                  ▀▀"
1020 PRINT"                                   ▀▀"
1030 PRINT"                                    ▀▀"
1040 PRINT"                                     ▀▀"
1050 PRINT"                                      ▀▀"
1060 PRINT"                                       ▀▀"
1070 PRINT"                                        ▀▀"
1080 PRINT"                                         ▀▀"
1090 PRINT"                                          ▀▀"
1100 PRINT"▀▀                                         ▀▀";
1110 PRINT"                                             ";
1120 PRINT"                ▌           ▐               ";
1130 PRINT"                ▌           ▐               ";
1140 PRINT"                ▌           ▐               ";
1150 PRINT"                ▌           ▐               ";
1160 PRINT" FUEL       >▌           ▐<    PUNTOS    ";
1170 PRINT"  500        ▌           ▐        0       ";
1180 PRINT"                ▌           ▐               ";
1190 PRINT"                ▌           ▐               ";
1200 PRINT"                ▌           ▐               ";
1210 PRINT"                ▌     ▌     ▐               ";
1220 PRINT"                ▌           ▐               ";
1230 PRINT"                ▌   RADAR▌  ▐               ";
1240 PRINT"                                             "
1250 POKE2023,160:POKE56295,9
1260 POKE2040,240:POKEV+39,1:POKEV,176:POKEV+1,79:POKEV+21,1
1300 RETURN
1500 REM ** FINAL DE JUEGO **
1505 POKEV+21,0:FORL=0TO24:POKESD+L,0:NEXT
1510 POKESD+14,5:POKESD+18,16:POKESD+3,1:POKESD+24,143:POKESD+6,240:POKESD+4,65
1515 FR=5389
1520 FORI=0TO70:POKE53265,PEEK(53265)AND239
1525 FQ=FR+PEEK(SD+27)*3.5:HF=INT(FQ/256):LF=FQ-HF*256
1530 POKESD,LF:POKESD+1,HF
1535 POKE53265,PEEK(53265)OR16
1540 NEXT I
1545 POKESD+24,0
2000 PRINT""
2002 PRINT"HAS CONSEGUIDO ";SC "PUNTOS"
2003 PRINT:PRINT"HAS DESTRUIDO ",SH "NAVES"
2025 IF HI<SC THEN HI=SC:PRINT"      NUEVO RECORD      "
2030 PRINT"RECORD=  ";HI
2100 FORX=0TO20:GETA$:NEXTX:Z=0:FORX=1TO5:IFSC>SC(X)THENZ=X:X=11
2105 NEXTX:IFZ=0THEN2140
```




```
2110 PRINT"▓▓▓▓▓▓ESCRIBE TU NOMBRE: "
2115 INPUTA$:IF LEN(A$)>10 THENA$=LEFT$(A$,10)
2120 IFZ=5THEN2130
2125 FORX=4TOZSTEP-1:SC(X+1)=SC(X):SC$(X+1)=SC$(X):SH(X+1)=SH(X):NEXTX
2130 SC(Z)=SC:SC$(Z)=A$:SH(Z)=SH
2140 PRINT"▓▓▓▓▓▓▓▓▓▓▓▓▓▓TABLA DE RECORDS"
2150 PRINT"▓▓▓▓▓▓▓▓▓▓▓▓▓▓────────────────────"
2151 PRINT"▓"TAB(5);"PUNTOS";TAB(13);"NA"
2152 PRINTTAB(5);"──────";TAB(13);"──"
2155 FORX=1TO5:PRINT"▓▓";X;TAB(5);SC(X);TAB(12);SH(X);TAB(18);SC$(X)
2160 NEXTX:X=FRE(0)
2240 PRINT:PRINT"▓▓▓▓▓OTRO JUEGO ? (S/N)"
2250 GETA$:IFA$=""THEN2250
2260 IFA$="S"THEN PRINT"▓":GOTO130
2270 IFA$="N"THEN PRINT"ADIOS":END
2280 IF A$<>"S"ANDA$<>"N"THEN2250
2500 REM ** TITULO **
2501 PRINT"▓▓****************************************";
2502 PRINT"****************************************"
2503 PRINT"[illegible]"
2504 PRINT"[illegible]"
2505 PRINT"[illegible]"
2506 PRINT
2507 PRINT"[illegible]"
2508 PRINT"[illegible]"
2509 PRINT"[illegible]"
2510 PRINT"▓▓▓▓          COMMODORE MAGAZINE"
2511 PRINT"▓▓▓ESPERA...▓"
2512 RETURN
3000 REM ** INSTRUCCIONES **
3002 PRINT"▓"
3004 PRINT"▓****************************************"
3006 PRINT"▓▓EL CONVOY ESTELAR ESTA SIENDO ATACADO"
3008 PRINT"POR NAVES ENEMIGAS DEL VACIO EXTERIOR."
3010 PRINT"TU HAS SIDO ENVIADO ABORDO DE UN"
3012 PRINT"CRUCERO ESTELAR PARA HACERLES FRENTE."
3014 PRINT
3016 PRINT"LA PRESENCIA Y POSICION DE LAS NAVES"
3018 PRINT"ENEMIGAS PUEDES VERLA EN TU RADAR"
3020 PRINT"DE ABORDO"
3022 PRINT"LAS DOS MARCAS DEL RADAR INDICAN"
3024 PRINT"TU DISTANCIA EFICAZ DE FUEGO."
3026 PRINT
3028 PRINT "CADA BLANCO QUE HAGAS, SUMARAS PUNTOS"
3030 PRINT"PERO SOLO DESTRUIRAS AL ENEMIGO SI"
3032 PRINT"LE ACIERTAS 5 VECES CON LO QUE"
3034 PRINT"APARECE UNA MARCA EN TU PANEL"
3036 PRINT"▓▓▓PULSA UNA TECLA▓"
3038 GETA$:IFA$=""THEN 3038
3040 PRINT"▓"
3042 PRINT"▓****************************************"
3044 PRINT"▓▓CONTROLA EL DESTRUCTOR CON TU JOYSTICK"
3046 PRINT"PERO RECUERDA QUE EL MOVIMIENTO DE TU"
3048 PRINT"NAVE A LA DERECHA SUPONE QUE LAS NAVES"
3050 PRINT"ENEMIGAS VAN A LA IZQUIERDA Y VICEVERSA"
3052 PRINT
```

PROGRAMA commodore COMPROBADO

```
3054 PRINT"EL CONVOY ESTELAR ESPERA QUE DESTRUYAS"
3056 PRINT"TANTAS NAVES COMO SEA POSIBLE, PERO ÑO"
3058 PRINT"DISPONES DE MUCHO COMBUSTIBLE Y CUANDO"
3060 PRINT"SE ACABE ACABARA TU MISION"
3062 PRINT"PULSA UNA TECLA Y EMPIEZA"
3064 GETA$:IFA$=""THEN3064
3066 RETURN
4999 REM ** ENTRADA DE SPRITES **
5000 FORS=15360TO15422:READQ:POKES,Q:NEXT
5001 FOR S=15424 TO 15486:READQ:POKES,Q:NEXT
5002 FORS=15488TO15550:READQ:POKES,Q:NEXT
5003 FORS=15552TO15614:READQ:POKES,Q:NEXT
5004 FORS=15616TO15678:READQ:POKES,Q:NEXT
5005 FORS=15680TO15742:READQ:POKES,Q:NEXT
5006 FORS=15744TO15806:READQ:POKES,Q:NEXT
5007 FORS=15808TO15870:READQ:POKES,Q:NEXT
6000 REM ** DATAS PARA VISTAS **
6003 DATA136,0,17,80,0,10,32,0,4,64,0,2,128,0,1,0,0,0,0,0,0,24,0,0,66,0,0
6004 DATA0,0,168,153,21,0,0,0,0,66,0,0,24,0,0,0,0,0,0,128,0,1,64,0,2,32,0
6005 DATA4,80,0,10,136,0,17
6010 REM ** DATAS NAVE PEQUENA **
6013 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,32,64,0,79,32,0,134,16,0
6014 DATA255,240,0,150,144,0,64,32,0,32,64,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
6015 DATA 0,0,0,0
6020 REM ** DATAS NAVE MEDIA-PEQUENA **
6023 DATA 0,0,0,0,0,0,0,0,9,242,0,24,163,0,56,67,128,112,225,192,161,80,160
6024 DATA159,95,32,159,95,32,160,160,160,112,65,192,56,3,128,24,3,0,8,2,0,0,0
6025 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
6030 REM ** DATAS NAVE MEDIA-GRANDE **
6033 DATA 0,0,0,0,0,0,0,0,4,0,128,12,252,192,28,72,224,56,48,112,112,120,56
6034 DATA160,180,20,159,183,228,160,180,20,159,183,228,162,73,20,114,49,56,56
6035 DATA 0,112,28,0,224,12,0,192,4,0,128,0,0,0,0,0,0,0,0
6040 REM ** DATAS NAVE GRANDE **
6043 DATA 0,0,0,2,0,64,6,0,95,14,126,112,30,36,120,60,24,60,120,126,30,144,153
6044 DATA 9,159,153,249,160,153,5,160,153,5,159,153,249,145,90,137,121,36,158
6045 DATA 60,24,60,30,0,120,14,0,112,6,0,96,2,0,64,0,0,0,0,0,0
6099 REM ** DATAS EXPLOSIONES **
6100 DATA 144,0,0,36,32,132,15,3,198,30,9,224,61,17,242,120,112,120,240,168,60
6101 DATA175,39,196,161,36,68,161,36,68,175,39,196,248,168,125,60,114,136,30,1
6102 DATA177,79,10,232,7,115,192,3,2,128,0,32,16,2,0,0,0,4,2,96,97,4
6110 DATA144,0,0,36,39,132,136,3,192,24,9,128,61,17,242,26,112,0,240,168,60,169
6111 DATA39,4,32,0,64,161,36,68,35,36,196,120,168,125,0,114,136,30,1,170,79,10
6112 DATA0,7,115,198,3,2,128,3,160,16,2,0,0,0,4,2,96,97,4
6120 DATA144,0,72,36,32,4,136,2,16,24,9,128,5,17,242,26,112,0,242,40,60,169,32
6121 DATA4,32,0,64,161,36,68,35,36,196,64,168,125,0,18,136,30,1,177,67,10,0,6
6122 DATA19,192,36,2,128,192,160,16,2,0,0,0,4,2,96,97,4
7000 RETURN
```

Concurso

Matrices y determinantes

Luis Pellicer Lorenzo, lector de Vigo, nos envía este interesante programa, de tipo matemático que él ha llamado **MAT & DET** y que permite realizar el producto de dos matrices de cualquier orden, siempre sin pasarse claro, ofreciendo también la posibilidad de cálculo de determinantes. Según datos de su autor, que hemos podido constatar, el programa es capaz de resolver un determinante de 7×7 en unos **4 segundos.** El programa calcula un determinante triangularizando la matriz correspondiente. Además no plantea ningún problema a la hora de introducir los datos, ya que, a través de una serie de explicaciones y preguntas, el programa guía completamente al usuario. Esperamos que todo este tipo de programas **científico-matemáticos** sean del agrado de nuestros lectores.

```
50000 REM################################
50010 REM##                            ##
50020 REM##"   ┐┘┘¯ ╱┤┌ ¯¯ ╲♠│ ─╮¯¯●   "##
50030 REM##"             &              "##
50040 REM##"  ¯♠L¯╱┘┌ ¯¯│¯─╲╱♠╱│¯     "##
50050 REM##"────────────────────────────"##
50060 REM##"   AUTOR: L╱● ┐┘┘╮¯¯─     "##
50070 REM##                            ##
50080 REM################################
50100 PRINT"♥"
50110 POKE53272,23:PRINTCHR$(8)
50120 PRINT"▮▮▶▶▶▶▶▶▶■**************************♥"
50130 PRINT"▶▶▶▶▶▶▶■***  ≡ │¯┘L¯♠ ┌¯╲┌╱   ■***♥".
50140 PRINT"▶▶▶▶▶▶■**************************♥"
50150 PRINT"▮▶▶◣1)  ≡RODUCTO ╲ATRICES "
50160 PRINT"▮▶▶◣2)  ≡ALCULO ¯ETERMINANTE♥"
50170 GETA$:IFA$=""THEN50170
50180 IFA$<>"1"ANDA$<>"2"THEN50170
50190 IFA$="2"THEN50700
50200 CLR
50210 REM******************************
50220 REM**                          **
50230 REM**  PRODUCTO DE MATRICES    **
50240 REM**                          **
50250 REM******************************
50260 PRINT"♥▮▮▶▶▶▶■**▶▶≡┐┘¯ ╱┤┌ ¯¯ ╲♠│ ─╮¯●▶▶■**♥"
50270 PRINT"◣▮▶▶,NTRODUCE ORDEN ╲ATRICES :"
50280 PRINT"▮▮◣┌RDEN ╲ATRIZ ♠ :"
50290 INPUT"▮▶▶◣─ILAS ≡";N:IFN=0THENPRINT"┐┐┐":GOTO50290
50300 INPUT"▮▶▶◣¯OLUMNAS ≡";M:IFM=0THENPRINT"┐┐┐":GOTO50300
50310 PRINT"▮▮◣┌RDEN ╲ATRIZ │ :"
50320 PRINT"▮▶▶◣─ILAS ≡?";M
50330 INPUT"▮▶▶◣¯OLUMNAS ≡";P:IFP=0THENPRINT"┐┐┐":GOTO50330
50340 DIMA(N,M),B(M,P),C(N,P)
50350 PRINT"▮▶▶◣ ╲NTRODUCE ELEMENTOS :▮▮"
50360 FORI=1TON:FORJ=1TOM
50370 PRINT"≡♥▶▶♠("I","J") = ";:INPUTA(I,J)
50380 PRINT"┐                                 "
50390 NEXTJ,I
50400 PRINT"◣┐▶▶ON CORRECTOS ( ● / / ) ?"
50410 GETA$:IFA$<>"S"ANDA$<>"N"THEN50410
50420 IFA$="N"THENPRINT"┐                                              ":GOTO50360
50430 PRINT"┐                                 "
50440 FORI=1TOM:FORJ=1TOP
50450 PRINT"≡♥▶▶│("I","J") = ";:INPUTB(I,J)
50460 PRINT"┐                                 "
50470 NEXTJ,I
50480 PRINT"◣┐▶▶ON CORRECTOS ( ● / / ) ?"
50490 GETA$:IFA$<>"S"ANDA$<>"N"THEN50490
```

```
50500 IFA$="N"THENPRINT"                                        ":GOTO50440
50510 PRINT"J"
50520 R1=TI:FORI=1TON
50530 IFI=N/2ORI=N/2+.5THENPRINTTAB(2)" - = ";
50540 PRINTTAB(7)" ";
50550 FORJ=1TOP
50560 FORS=1TOM
50570 C(I,J)=A(I,S)*B(S,J)+C(I,J)
50580 NEXTS
50590 C(I,J)=INT(C(I,J)*1E4)/1E4
50600 PRINTC(I,J);
50610 NEXTJ
50620 PRINT" "
50630 NEXTI:R2=TI:PRINT"IEMPO MPLEADO : "(R2-R1)/60"SEG."
50640 PRINT"";
50650 GETA$:IFA$=""THEN50650
50660 RUN
50700 REM******************************
50710 REM**                          **
50720 REM**  CALCULO DETERMINANTE    **
50730 REM**                          **
50740 REM******************************
50750 CLR
50760 PRINT"**"
50770 INPUT"NTRODUCE ORDEN ETERMINANTE : ";N
50780 IFN=0THENPRINT"                                  ":GOTO50770
50790 DIMA(N,N),S(N)
50800 PRINT"NTRODUCE ELEMENTOS :"
50810 FORI=1TON:FORJ=1TON
50820 PRINT"("I","J") = ";:INPUTA(I,J)
50830 PRINT"                         "
50840 NEXTJ,I
50850 PRINT"ON CORRECTOS (  / / ) ?"
50860 GETA$:IFA$<>"S"ANDA$<>"N"THEN50860
50870 IFA$="N"THENPRINT"                                        ":GOTO50810
50880 PRINT"                         "
50890 R1=TI
50900 FORI=1TON-1
50910 FORT=I+1TON
50920 B=A(I,T):C=A(I,I)
50930 IFB=0ANDC=0THEN50980
50940 IFC=0THENGOSUB51100:GOTO50900
50950 FORS=ITON
50960 A(S,T)=A(S,T)-(B/C)*A(S,I)
50970 NEXTS
50980 NEXTT,I
50990 L=1:FORR=1TON
51000 L=A(R,R)*L
51010 NEXTR
51020 R2=TI:PRINT"OLUCION : "((-1)↑D)*L
51030 PRINT"IEMPO EMPLEADO :"(R2-R1)/60"SEG."
51040 PRINT"";
51050 GETA$:IFA$=""THEN51050
51060 RUN
51100 D=D+1
51110 FORK=1TON
51120 S(K)=A(K,T-1):A(K,T-1)=A(K,T)
51130 A(K,T)=S(K)
51140 NEXT:RETURN
```

PROGRAMA commodore COMPROBADO

Concurso

Dibujo con joystick

Daniel Cocho González, lector de **Commodore Magazine** en Madrid, nos remite un interesante programa que él ha llamado DIBUJO JOY, nombre muy adecuado para este programa, ya que permite realizar dibujos con el *joystick*, sobre la pantalla de alta resolución del **64.** El programa comienza presentando en pantalla las instrucciones para la realización de los dibujos. Estas instrucciones son sencillas; para dibujar una línea basta mover el *joystick* en la dirección deseada, que puede ser horizontal, vertical o diagonal. En cualquier momento, se puede pulsar el botón de disparo del *joystick* lo que equivale a "levantar el lápiz del papel", de esta forma el punto deja de imprimir, aunque puede seguir moviéndose por la pantalla.

Después de las instrucciones, el programa pide las coordenadas X, Y, para situar el punto inicialmente sobre la pantalla. A partir de aquí todo depende del dibujante, que con un poco de práctica podrá comprobar cómo queda, sobre la pantalla de alta resolución, cualquier dibujo que se le ocurra.

```
5 REM  ************************
10 REM * DIBUJO A/R JOYSTICK *
15 REM *                    *
20 REM * DANIEL COCHO (1984) *
25 REM ************************
26 GOSUB450:GOSUB750
27 REM
28 REM ** PREPARACION DE LA A/R **
29 REM
30 PRCP=2*4096:POKE53272,PEEK(53272)OR8
35 PRCP=2*4096:POKE53272,PEEK(53272)OR8
40 POKE53265,PEEK(53265)OR32
50 FORI=PRCPTOPRCP+7999:POKEI,0:NEXT
60 FORI=1024TO2023:POKEI,5:NEXT
65 REM
66 REM ** COLISION CON EL BORDE **
67 REM
70 IFX<0THENX=0
71 IFX>319THENX=319
72 IFY<0THENY=0
73 IFY>199THENY=199
74 REM
75 REM ** VA A LEER EL JOY **
76 REM
80 GOSUB400
90 GOTO600
397 REM
398 REM ** LEE EL JOYSTICK **
399 REM
400 JV=PEEK(56320)
410 FR=JVAND16
420 JV=15-(JVAND15)
430 RETURN
439 REM
440 REM ** PRESENTACION INICIAL **
441 REM
450 PRINT"♥":PRINT:PRINT:PRINT:PRINT"▌       DIBUJO EN A/R CON EL JOYSTICK"
460 PRINT:PRINT:PRINT:PRINT:PRINTTAB(6)"▐EL JOYSTICK DESPLAZA EL PUNTO"
470 PRINT:PRINTTAB(6)"POR LA PANTALLA.EL BOTON PARA"
480 PRINT:PRINTTAB(6)"DISPARAR IMPIDE LA IMPRESION"
490 PRINT:PRINT:PRINT:PRINT:PRINTTAB(6)"▜PULSA UNA TECLA PARA EMPEZAR"
491 GETA$:IFA$=""THEN491
495 PRINT"♥"
496 RETURN
497 REM ** RUTINA DE IMPRESION **
498 REM
500 QQ=INT(X/8)
510 WW=INT(Y/8)
```

```
520 JJ=YAND7
530 YY=PRCP+WW*320+8*QQ+JJ
540 XX=7-(XAND7)
550 POKEYY,PEEK(YY)OR(2↑XX)
560 RETURN
569 REM
570 REM ** MUEVE EL PUNTO **
571 REM
600 IFJV=1THENY=Y-1:REM *ARRIBA*
610 IFJV=2THENY=Y+1:REM *ABAJO*
620 IFJV=3THEN70:REM *SIN EFECTO*
630 IFJV=4THENX=X-1:REM *IZQUIERDA*
640 IFJV=5THENY=Y-1:X=X-1:REM *IZQ-ARR*
650 IFJV=6THENY=Y+1:X=X-1:REM *IZQ-ABA*
660 IFJV=7THEN70:REM *SIN EFECTO*
670 IFJV=8THENX=X+1:REM *DERECHA*
680 IFJV=9THENY=Y-1:X=X+1:REM *DER-ARR*
690 IFJV=10THENY=Y+1:X=X+1:REM *DER-AB*
691 REM
692 REM ** DECIDE LA IMPRESION O NO **
693 REM
695 IFFR<>16THEN70
700 GOSUB500
705 FORT=1TO5:NEXT
710 GOTO70
719 REM
720 REM ** PUNTO AL ORIGEN DESEADO **
721 REM
750 PRINTTAB(3)"▮INTRODUCE LAS COORDENADAS DEL ORIGEN":PRINT
760 PRINTTAB(9)"EN EL QUE VAS A EMPEZAR":PRINT:PRINT:PRINT
770 PRINT:INPUT"▮     X  (DE 0 A 319)";X
780 PRINT:INPUT"      Y  (DE 0 A 199)";Y
790 PRINT:PRINT:PRINT:PRINT:PRINTTAB(3)"▮EL JOYSTICK DEBE SER CONECTADO AL"
800 PRINT:PRINTTAB(16)"▮PORT 2"
810 PRINT:PRINT:PRINT:PRINT:PRINTTAB(12)"▮PULSA UNA TECLA"
820 GETB$:IFB$=""THEN820
830 PRINT"▮"
840 RETURN
```

PROGRAMA commodore COMPROBADO

Interface RS 232 para el VIC-20

Supongamos que se presenta la oportunidad de conectarle a nuestro **Vic 20** una impresora que utilice el *interface* estándar RS-232. Serían varias las posibles soluciones, pero la que aportamos en este artículo es tan válida como cualquier otra. Consiste simplemente en fabricarnos nuestro propio *interface*, que posibilita que el ordenador se ponga de acuerdo con la impresora.

Básicamente, la función del circuito consiste en normalizar los niveles de tensión de las señales. El montaje se divide en dos circuitos. El primero es la fuente de alimentación, que proporciona las tensiones adecuadas para que funcione el *interface*. El segundo es el *interface* propiamente dicho.

Después de haber sido comprobado con diversas impresoras, el funcionamiento fue óptimo. Algunos de los modelos utilizados en la prueba son la IDS Paper 440, Anadex DP8000, Decwriter LA 120, etc., pero trabajará igualmente con otros modelos.



FUNCIONAMIENTO CLASICO

En muchos microordenadores se ha dispuesto internamente un dispositivo conocido como **UART** *(Universal Asyncronus Receiver Transmiter)*, para encargarse de las tareas relacionadas con las comunicaciones con el mundo externo. Físicamente, la **UART** es un circuito integrado que se complementa con otros *chips*, que disponen de buffers, encargados de normalizar los niveles de las tensiones en las líneas.

Cuando en un circuito los *chips* utilizados corresponden a la tecnología TTL (Transistor-Transistor-Logic), los niveles de tensión que aparecen son siempre 0 ó +5 voltios. Sin embargo, para la transmisión de los datos se requieren niveles de +0 y —10 voltios, por los que es necesario realizar la conversión.

Para terminar de matizar sobre la **UART,** digamos que encuentra entre sus posibilidades la disposición de los datos que llegan en serie, un bit después de otro a paralelo, acumulando el número necesario de bits para constituir el byte a su salida. Igualmente puede hacer lo contrario, recoger varios bits simultáneamente e ir transmitiéndolos uno a uno, por orden. También temporiza la entrada y salida de datos, de tal manera que la transferencia se efectue a una determinada velocidad. Paralelamente genera diversas señales de control, que facilitan el "apretón de manos", que quiere decir que siendo afirmativo la comunicación del dato ha sido correctamente recibida.

En el momento en que se inició la fabricación del **Vic 20, Commodore** no pudo disponer de la **UART** que habría requerido, por lo que optó por emular mediante *software* el *interface.* El *chip* utilizado fue un clásico llamado **VIA 6522** *(Versatile Inteface Adaptor).* un dispositivo extremadamente complejo que describiremos en un futuro artículo.

El usuario puede acceder directamente a las líneas del **VIA,** por medio de *port* de usuario del **Vic 20.** Sin embargo, los niveles de tensión utilizados corresponden al estándar TTL.

Esto imposibilita la conexión directa de la impresora. Por lo tanto, necesitaremos algún tipo de *buffer* que solvente los problemas de adaptación de los niveles. En el montaje que presentamos se incluye la circuitería necesaria.

Otro importante factor a tener en cuenta es la velocidad a que puede trabajar la impresora, que principalmente queda limitada por los componentes mecánicos de la máquina. Como es lógico, la velocidad con que afluyen los datos es mucho mayor que la velocidad con que se pueden imprimir físicamente los caracteres. A partir de aquí volvemos a encontrarnos con la palabra *buffer*, pero en este caso su significado es distinto. Viene a ser una memoria tampón en forma de memoria RAM, que va dispuesta en el interior de la impresora. En ella se almacenan los datos que serán impresos. Mientras exista espacio vacío en el *buffer* la impresora seguirá aceptando datos. Para proporcionarle esta información al ordenador existe una línea llamada DTR *(Data Terminal Ready).* Normalmente los sensores de final de papel también están conectados a la línea DTR.

En un *interface* sencillo, DTR se conecta a la línea CTS *(Clear To Send)* del ordenador. Cuando se llena el *buffer*, la impresora pone DTR en estado bajo. Cuando el ordenador observa que la señal en CTS está en el nivel bajo, deja de enviar datos a la salida hasta que vuelve a ponerse en estado alto. Aunque suena prometedor, es una pena que el *interface* RS-232 para el **Vic 20** no trabaje así.

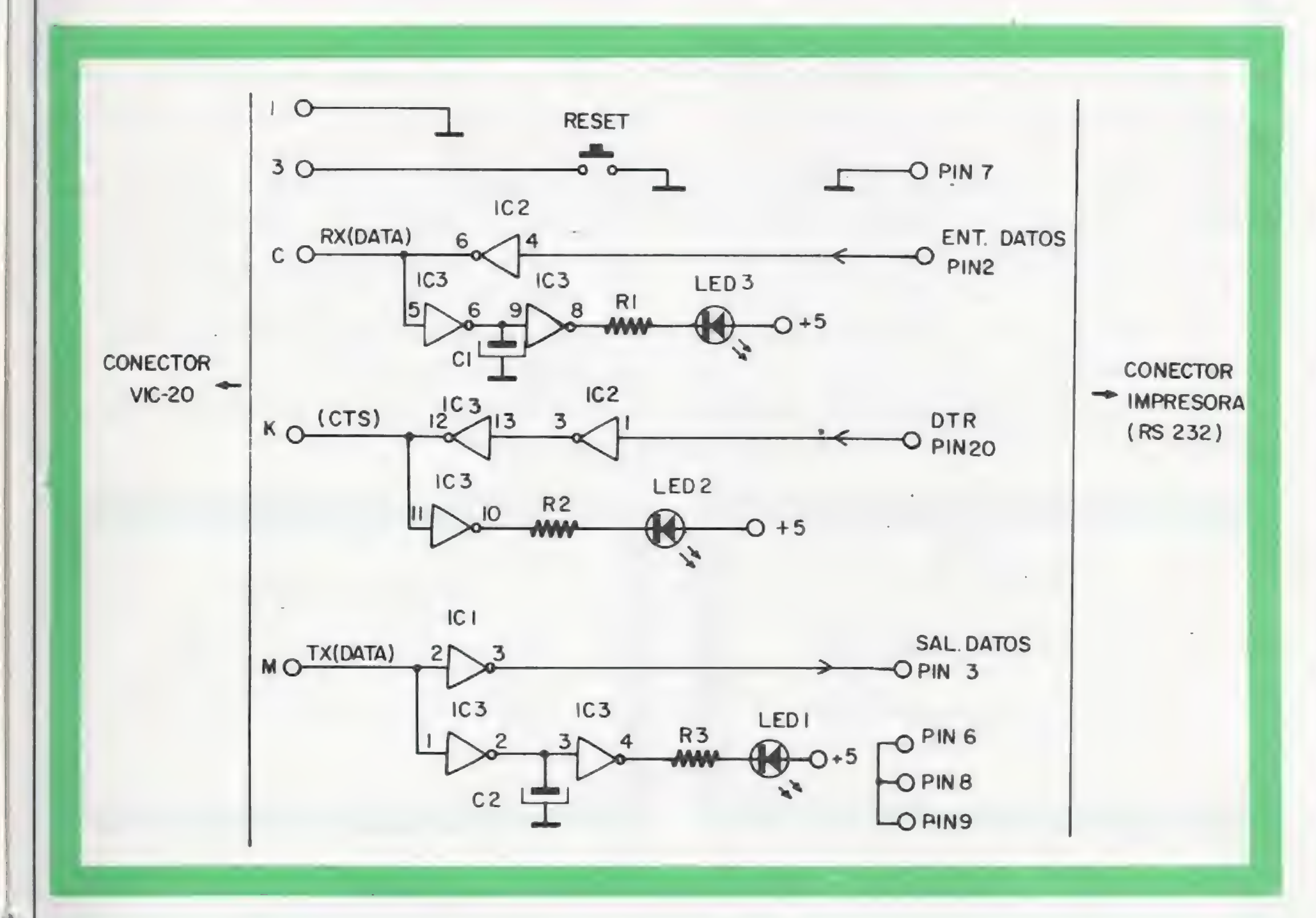


Figura 1. Esquema teórico del *interface* RS-232.

El conjunto de señales DTR, CTS, RTS, DCD, etc., además de la información correspondiente, es quien se encarga de efectuar el apretón de manos. El único problema es que el sistema operativo del **Vic** no comprueba el CTS. Si se establece el apretón de manos, el sistema operativo tampoco funciona de esta forma. La causa es simplemente que el *software* necesario no está presente en la máquina.

Por esta razón, el *software* y el *interface* que describiremos trabajan como un sistema de tan solo dos líneas; transmisión de los datos y masa. La recepción de datos hacia el ordenador también ha sido prevista para quienes deseen alimentar al **Vic** con datos. El DTR ha sido incluido por dos razones. En primer lugar, alguien podría querer escribir su propio programa en código máquina, para manejarla. En segundo se podría detectar que algo ocurre observando que el piloto de testigo conectado a DTR está apagado, para que pongamos la impresora en orden.

La velocidad de transferencia de datos adoptada es de 600 baudios, lo suficientemente rápida para la mayoría de las aplicaciones, sin que cause problemas con el DTR. Si la impresora en cuestión no pudiera adaptarse

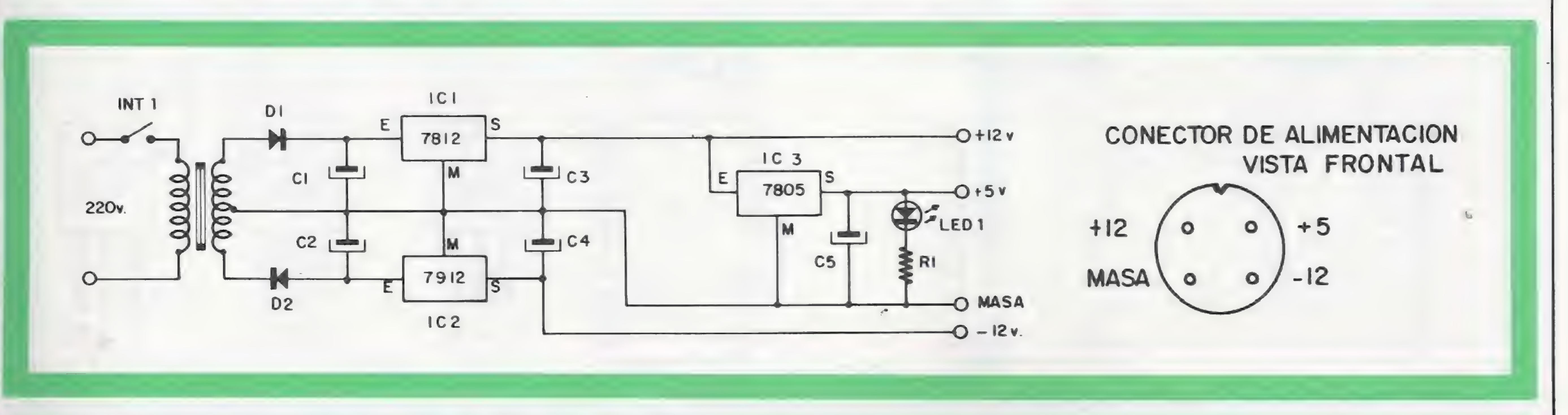


Figura 2. Esquema teórico de la fuente de alimentación.

a ella, se puede reducir a 300 baudios. También se puede hacer que el **Vic** vaya más deprisa, pero si la impresora recibe más datos que los que caben en su *buffer*, se perderán algunos.

Todo lo que se requiere para controlar un dispositivo RS-232 desde un **Vic 20** es algo de *hardware*, que convierta los niveles TTL del **Vic** a otros de más y menos 10 voltios.

Los circuitos integrados controladores de líneas utilizados son los MC1488 y MC1489, que proporcionan las tensiones de línea adecuadas. Se utiliza también un tercer *chip*, el 7404, que es un sextuple inversor.

Los condensadores C7 y C8 reducen la velocidad de parpadeo de los LEDs, para que sean más fáciles de visualizar. Cuando se está desarrollando un programa en el que se deberá utilizar impresora, pero no está en la parte del problema en la que se trabaja, el LED que indica la transmisión de datos es muy útil, para indicar una operación de impresora sin necesidad de tenerla conectada.

La segunda parte del montaje es la fuente de alimentación. El transformador reduce la magnitud de la tensión de la red. Posteriormente se realiza una rectificación de media onda con cada uno de los diodos. La disposición de cada uno de ellos produce que se obtenga tensión positiva los terminales de un condensador y tensión negativa en los del otro. A continuación, los *chips* estabilizadores de tensión 7812 y 7912, permiten disponer a sus respectivas salidas de +12 y -12 voltios. La tensión de +5

voltios, necesaria para alimentar el 7404, se obtiene a partir de otro *chip* similar, el 7805.

El botón de Reset no tiene ninguna misión especial en el montaje, simplemente se ha añadido para poder inicializar el ordenador sin volver a él.

Ambos circuitos han sido alojados en respectivas cajas de plástico, fácilmente adquiribles en el comercio. El cable que conecta con la impresora termina en un conector tipo D de 25 patillas, aunque se puede utilizar otro tipo adecuado. Para la conexión al Vic se recurre al conector clásico, que se ajuste a la configuración del *port* de usuario.

Los circuitos impresos a utilizar son los que aparecen en las figuras anexas, aunque cada cual puede rediseñarlo como guste.

Los circuitos integrados MC1488 y 1489 tienen sus equivalentes en las versiones 75188 y 75189 de **Texas Instruments,** por si hubiera dificultad en lozalizar los primeros. Una vez finalizado el montaje, si no hay impresora conectada, solamente lucirá el LED de encendido. Si se encendiera algún otro habrá que verificar, pues existirá un error en el montaje. Después se conecta la impresora, se activa su interruptor de alimentación, encendiéndose el LED correspondiente a DTR. Si hasta aquí está todo bien, el *interface* estará listo para su manejo.

Ahora sólo queda preparar el *software* adecuado para que se produzca la impresión. Existe un número de características que sufren ligeras variaciones, según el modelo de la impresora. Se dispondrá la impresora de acuerdo con las características proporcionadas por el fabricante, de la siguiente manera:

- Opciones de paridad-seleccione sin paridad.

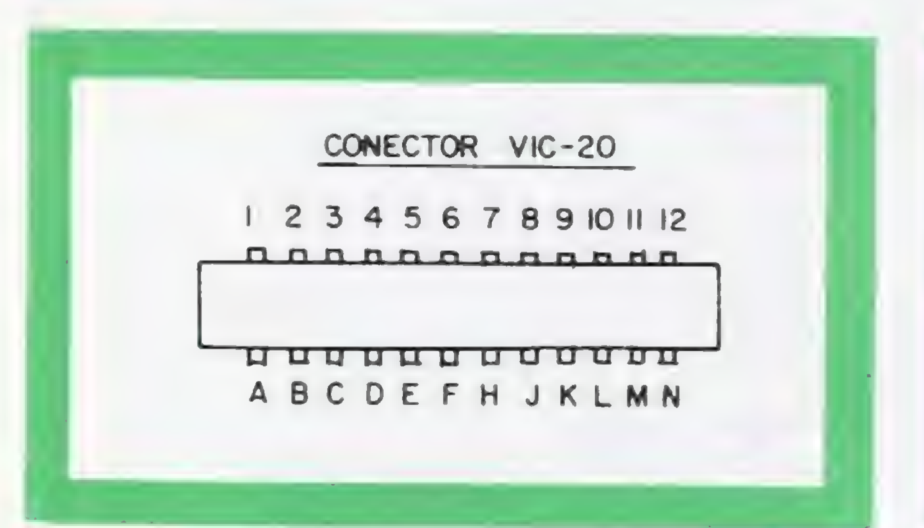


Figura 4. *Port* de E/S del **Vic 20.**

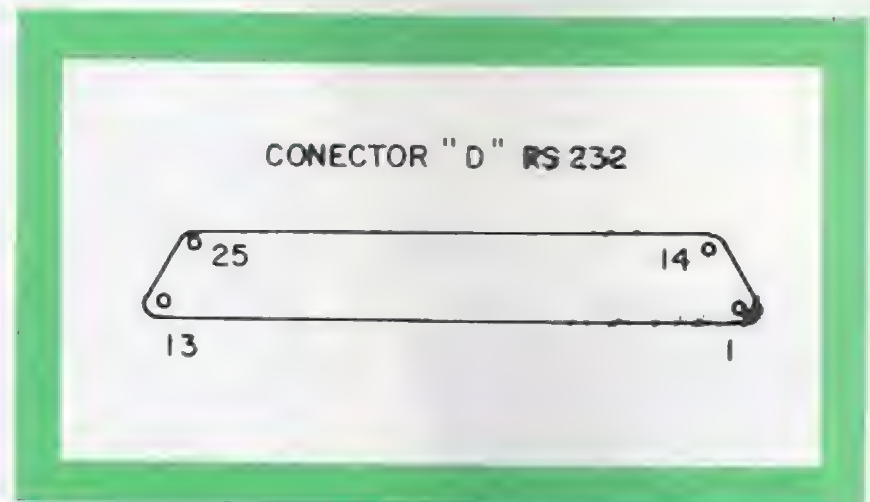


Figura 5. Diagrama del conector RS-232 tipo D.

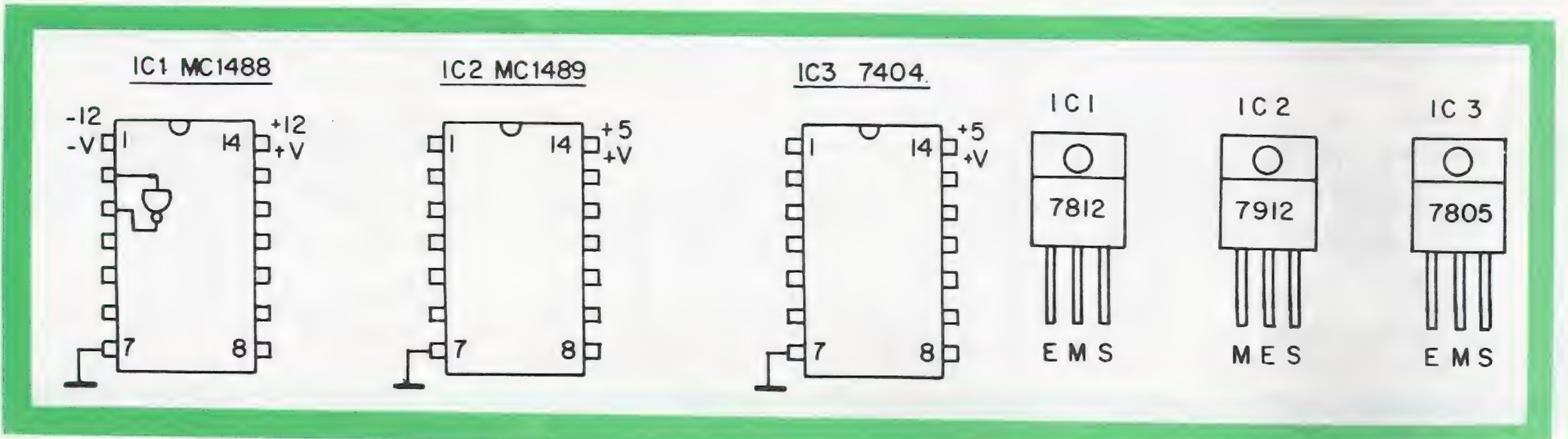


Figura 3. Diagrama de patillas de los circuitos integrados.

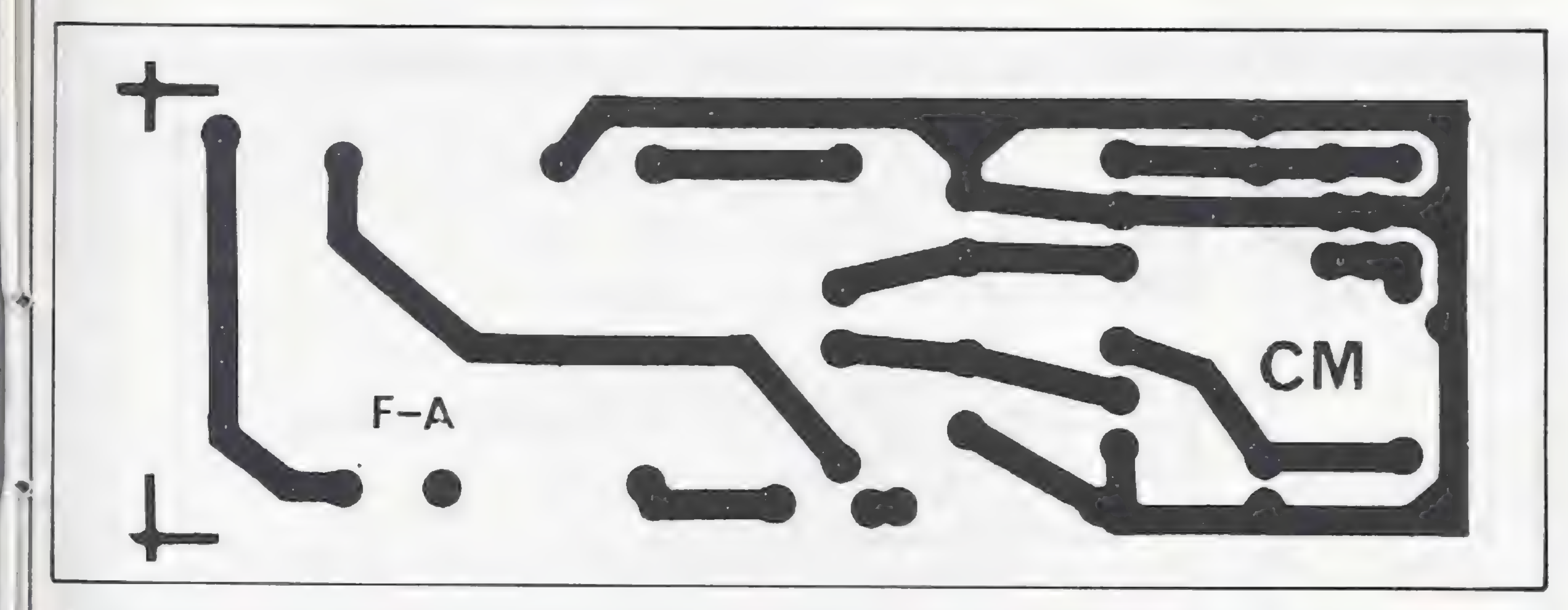


Figura 8. Circuito impreso para la fuente de alimentación.

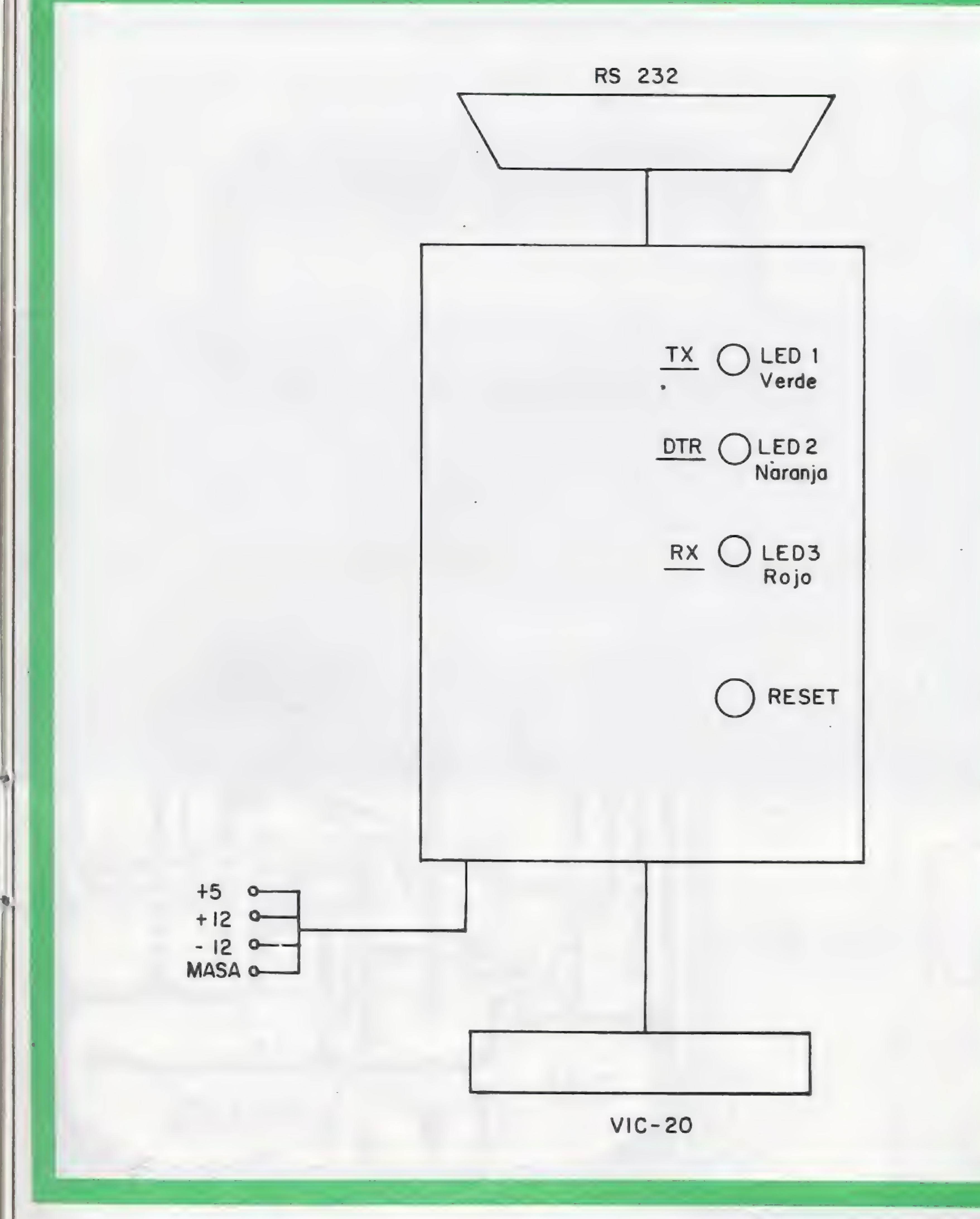


Figura 6. Conexión entre dispositivos.

● Palabras de 7 u 8 bits-seleccione palabras de 8 bits.

● RS-232 o bucle de corriente-seleccione RS-232.

● Alimentación automática de línea-seleccione on consulte el *software* de más adelante).

— Códigos de control de la impresión-seleccione off.

— Caracteres por línea-seleccione 80.

— Modo carácter/línea-seleccione 80.

— Xon/Xoff-seleccione off.

— Velocidad de transferencia-seleccione 600 baudios.

Para que la **VIA** del ordenador actue como un *port* RS-232, será necesario hacer una definición en dos registros. Estos con "Control", para la velocidad de transferencia, la longitud de la palabra, bits de Stop y "Comando", para la paridad, el apretón de manos, duplex y marca o espacio transmitidos. Disponiendo la impresora como ya se ha dicho, no se necesita el registro de Comando. El registro de control está localizado en la dirección decimal 659. Se puede definir mediante un POKE si se requiere. Una vez que se ha verificado que la impresora está con las características definidas, se teclea: OPEN 2, 2, 0, CHR$ (7), con alimentación de línea automática, 600 baudios y palabra de 8 bits. También se podría teclear: OPEN 2, 2, 0: POKE 659, 7.

El primer número que aparece tras OPEN es el número del fichero lógico. Si es menor de 127, el ordenador no generará alimentación de línea. Puesto que sólo podemos abrir (OPEN) un *port* RS-232, es más fácil utilizar 2 ó 200 como números de fichero, pero tampoco hay porque ceñirse a estos números. Después de introducir el comando de apertura, no habrá ocurrido nada que pueda verse, pero dentro del ordenador han ocurrido muchas cosas. Lo primero que ha hecho ha sido realizar un CLR. Si la instrucción de apertura se ejecuta en un programa después de cualquier DIM o sentencia con variables, se perderá. Abrir siempre el *port* RS-232 al comienzo del programa, asegurándonos que el programa no puede hacer un bucle hacia atrás



hasta allí. También se comprobará que se han perdido 512 bytes de memoria. Son tomados por el Vic a partir del límite superior a la zona de memoria destinada al BASIC, proporcionando *buffers* de 256 bytes para transmisión y recepción de datos. Si previamente hubiera algún programa allí, se perdería. Por tanto, hay que asegurarse de que queda suficiente memoria para el *port*.

Una vez en este punto, compruebe que existe DTR en el *interface*, teclee CMD2. La impresora deberá responder con "READY". Ya se puede teclear cualquier cosa de hasta una longitud de 80 caracteres, destinada a la impresora.

Aníbal Pardo

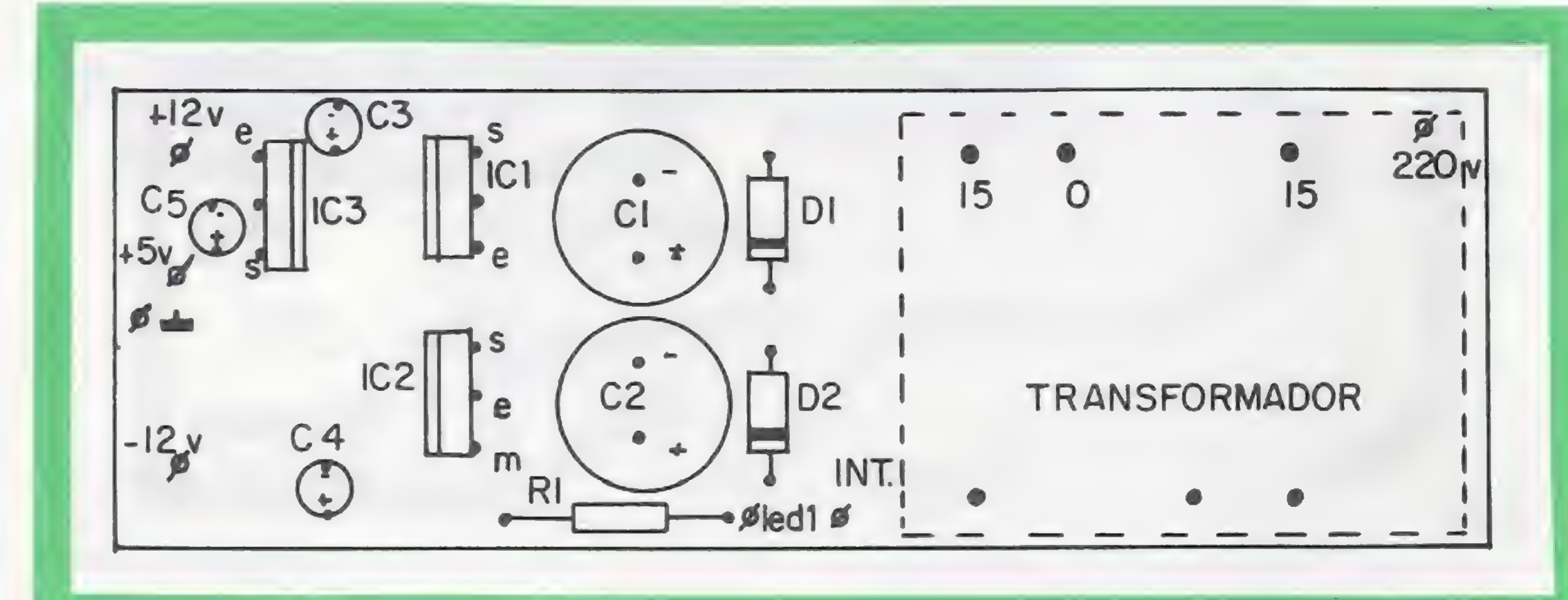


Figura 10. Montaje de la fuente.



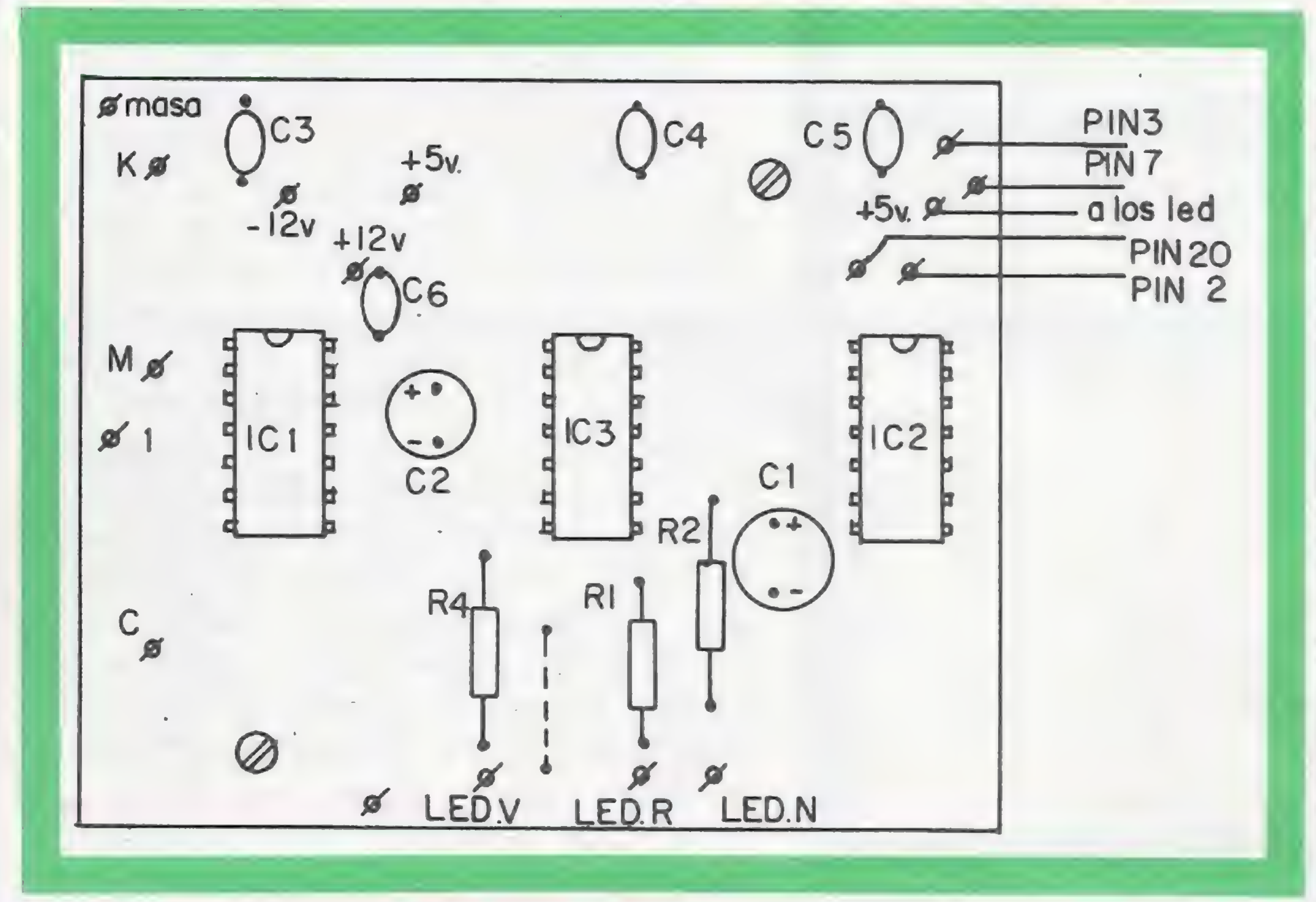


Figura 9. Montaje del interface.

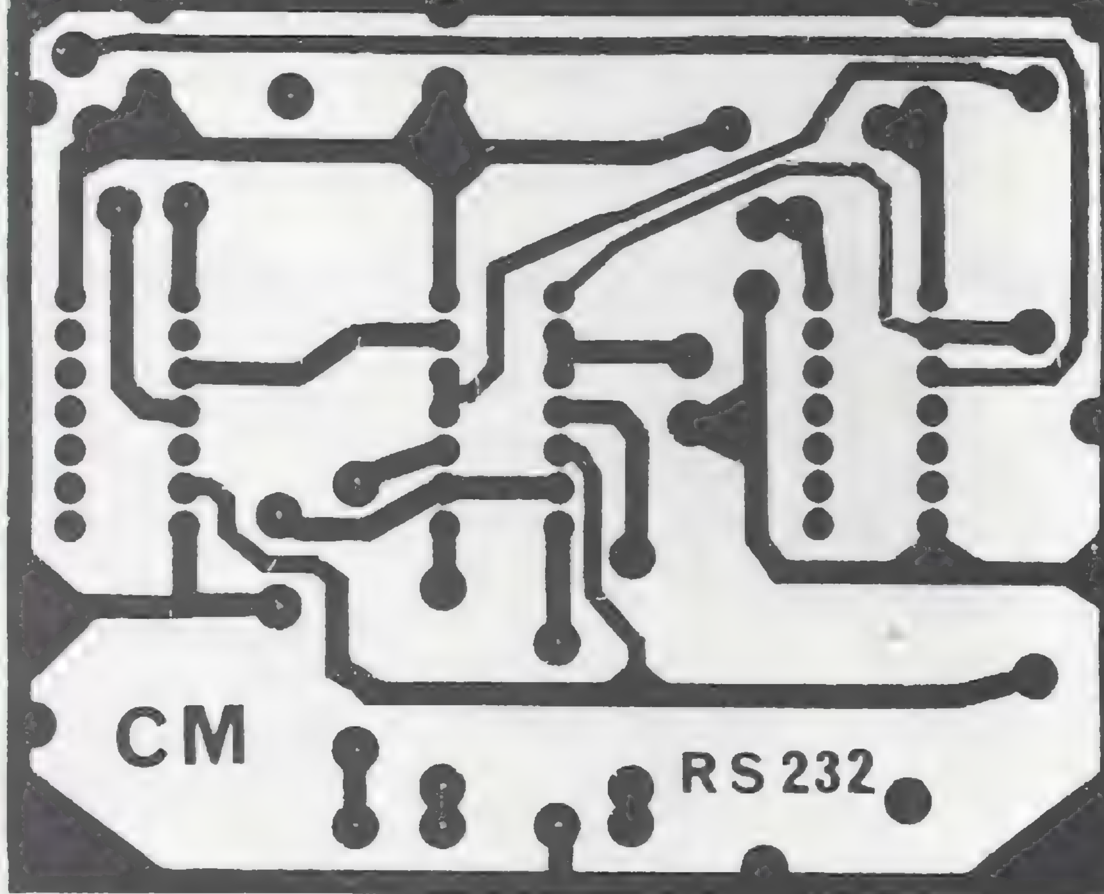


Figura 7. Circuito impreso para el *interface*.

FUENTE DE ALIMENTACION

LISTA DE MATERIALES

RESISTENCIAS

Rl = 560 ohm 1/2 w

CONDENSADORES

C1, C2 = 100μf 63v
C3, C4, C5 = 1μf 35v Tantalo

SEMICONDUCTORES

D1, D2 = 1N4004
IC1 = MC7812
IC2 = 7912
IC3 = 7805

VARIOS

1 Caja para montaje
1 Transformador 220 v/15+15 300 mA
1 Conector de cuatro contactos (Macho y Hembra)
1 metro de cable paralelo con clavija
1 interruptor
1 LED rojo

INTERFACE

LISTA DE MATERIALES

RESISTENCIAS

R1 = R2 = R3 = 220 ohm 1/4 w

CONDENSADORES

C1 = C2 = 47μf 16v
C3 = C4 = C5 = C6 = 100 kpf

SEMICONDUCTORES

IC1 = MC1488
IC2 = MC1489
IC3 = 7404

VARIOS

3 LED (Rojo, Naranja, Verde)
1 Conector "D" de 25 pines
1 Conector para el VIC 20 de 12 pines
1 Caja para montaje
3 Zócalos dual-Line de 14 pines
1 Pulsador para RESET





Cómo diseñar juegos para ordenador (capitulo 3)



Los juegos de aventuras

Básicamente, los juegos de aventuras son programas que intentan emular situaciones reales (o casi reales): buscar un tesoro en una isla, encontrar un asesino, etc. Son juegos en los que el jugador tiene que desarrollar dotes de observación e ingenio mental para poder terminarlos. La persona se sirve del ordenador como si fuese un vulgar sirviente que le desplaza, mire cosas, coja objetos y realice otras acciones de diversa índole. Pero el ordenador también le pone dificultades, no ya como el sirviente que hemos visto antes, sino en forma de sucesos externos. Si nos estamos desplazando por una casa nos podrá decir que no podemos pasar por una puerta, que está cerrada con llave, o que al subir unas escaleras se han caído y nos hemos matado. Estos programas proporcionan horas de diversión y no son excesivamente difíciles de realizar, constituyendo una introducción de indudable atractivo al mundo de los juegos.

Partes del juego

Una vez que hemos decidido crear un juego de aventuras hay que definir las distintas partes que lo forman. La primera debe ser el lugar donde sucede la acción. Una casa, una caverna, una isla pueden servir perfectamente para nuestro propósito. Evidentemente, sería mucho más interesante disponer de un escenario ilimitado, en el que nos pudiésemos desplazar a cualquier sitio, pero la memoria necesaria también sería ilimitada, por lo que tendremos que olvidar la idea. En nuestro caso vamos a elegir un castillo abandonado (con la excepción de alguna sorpresa que podamos encontrar) con dos pisos y seis habitaciones en cada piso. En este momento se debe tomar la primera decisión, podemos elegir entre dejar una configuración del castillo fija (las puertas y escaleras en el mismo sitio, aunque no los demás objetos) o variable, de modo que nunca se pueda saber cómo es el castillo. Por sencillez de diseño, elegiremos una configuración fija, definida en la figura 1.

A continuación hay que definir al jugador, en algunos juegos es una sola persona que tiene unas características que van cambiando con el tiempo (fuerza, agilidad, inteligencia, experiencia), mientras que en el otro sistema cada orden que damos se refiere a un grupo de personas —comandadas por nosotros— que sufren diversa suerte, según su estado, en este caso nos podemos encontrar con situaciones en las que tengamos 20 hombres y a los tres minutos, después de una pelea, sólo tengamos cinco. Este segundo caso es mucho más completo y difícil de manejar. Los pertrechos que podemos llevar dependen del número de personas que haya. La comida varía con la misma proporción, etc. Nosotros optaremos por la simplicidad y supondremos que el "aventurero" está solo.

Una vez definido el escenario y el jugador, le toca el turno a los objetos y bichos que estén en el castillo. Unos serán beneficiosos; otros, peligrosos, y algunos no tendrán ningún objeto (excepción hecha de la función de despistar). Incluso pueden tener datos u objetos ocultos, que no se ven hasta que lo miremos bien. En nuestro caso vamos a definir los siguientes objetos:
MONEDA MAGICA

ARCON ENCANTADO
MONSTRUO DE LAS GALLETAS
COLLAR DE LOS DIOSES
LIBRO
REVISTA COMMODORE MAGAZINE
CAJA DE GALLETAS
CUADRO ANTIGUO

Algunos objetos tienen una misión específica, como es el caso de la moneda mágica, que nos sirve para salir del castillo y del monstruo de las galletas con una misión que no conviene revelar ahora (ya la descubirá algún inocente), mientras que el libro, la revista y el cuadro pueden tener misiones distintas cada vez que se juegue.

A continuación nos toca definir las acciones que podemos realizar, el mínimo número de acciones que toda aventura que se precie debe tener consiste en:

Ve: sirve para moverse en una dirección, normalmente (como en nuestro caso) va seguida de un punto cardinal.

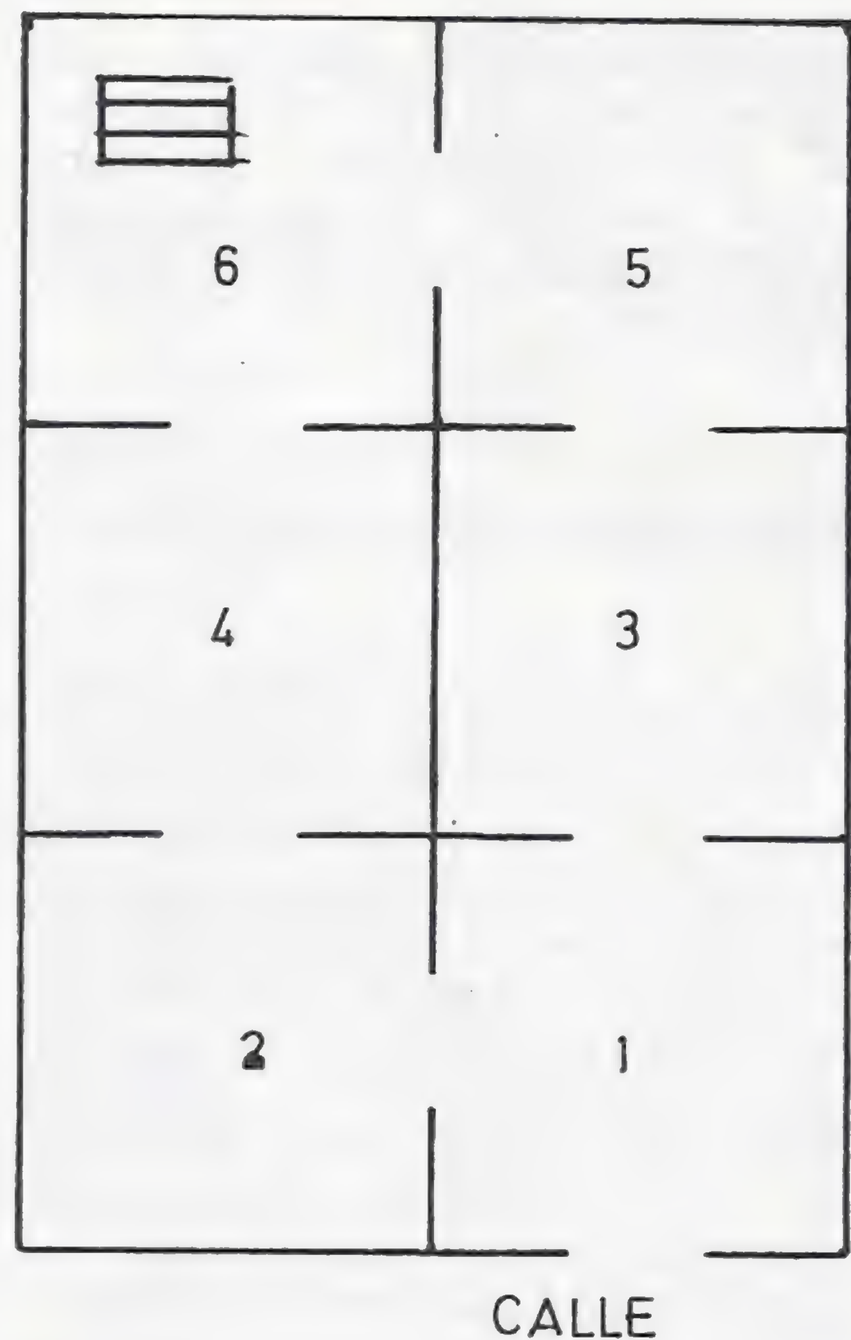
Haz: suele ir seguida de la palabra INVENTARIO y nos dice las cosas que llevamos encima.

Mira: seguida de un objeto nos da una descripción más detallada de él. En algunas aventuras también se acepta MIRA HABITACION que nos da una descripción de la habitación en sí. En este caso no es necesario, ya que esta descripción la da el programa automáticamente.

Coge: evidentemente coge el objeto que le indiquemos (si se puede coger).

Deja: deja un objeto que llevemos. La mayoría de los programas (y éste no es una excepción) le permite llevar un peso máximo, por lo que no podrá coger todos los objetos que desea.

Los demás comandos que se añadan serán suplementarios y no tendrán tanta importancia. Nuestro programa va a tener uno más (imprescindible en este caso): **Di,** que deberá ir seguido de la palabra CONJURO para tener algún efecto interesante, pero eso se debe a necesidades del guión y no suele estar incluido con los vocabularios normales de los juegos.



PISO BAJO

12 11
10 9
8 7

PISO ALTO

El resto de los comandos más o menos usuales son:

Abre: si hay puertas o cajas cerradas necesitaremos abrirlas (con llave la mayoría de las veces) para poder ver lo que hay dentro.

Mata, ataca: cualquier frase de este tipo nos impelerá a matar a nuestro enemigo, aunque los resultados no sean satisfactorios, en este juego, por desgracia, somos pacíficos y no atacamos a nadie.

Salva/Lee: si el juego que usted diseñe es muy largo, se encontrará en la disyuntiva de no dormir o de no terminarlo. Estos dos comandos dan la posibilidad de salvar el estado del juego y poder seguir jugando el día siguiente.

Aparte de estos comandos, cada juego puede disponer de los suyos propios que se dejan a la imaginación del programador, sin embargo, debe tenerse en cuenta que la persona que juegue disfrutará si comprende el juego y lo sabe manejar. No hay nada más frustrante que el saber qué hacer, pero no saber cómo decírselo a la máquina, el diccionario de verbos (y el de sujetos que veremos a continuación) deben ser claros y sencillos.

Una vez que hemos visto los verbos o acciones que podemos utilizar, vamos a ver los sujetos que se pueden usar con ellos. Toda frase que se dé al ordenador debe ir compuesta de un verbo y un sujeto, evidentemente se podría complicar la cosa de modo que pudiésemos decir:

VE NORTE

para que el programa nos comprenda y ejecute la acción.

Las palabras que pueden acompañar a un verbo son de muy diversa índole, en primer lugar los cuatro puntos cardinales:

NORTE
SUR
ESTE
OESTE
ARRIBA
ABAJO

son necesarias para el movimiento. Gran cantidad de acciones se refieren a los objetos definidos anteriormente, por lo que pasan a formar parte de nuestro diccionario; en algunos casos, abreviadas; el monstruo de las galletas será mencionado por el jugador simplemente como MONSTRUO y el

colar de los dioses pasa a ser COLLAR, de modo que podemos decir:
COGE COLLAR

El último grupo de palabras tienen muy diverso significado y comprende a las siguientes:
INVENTARIO
CONJURO...

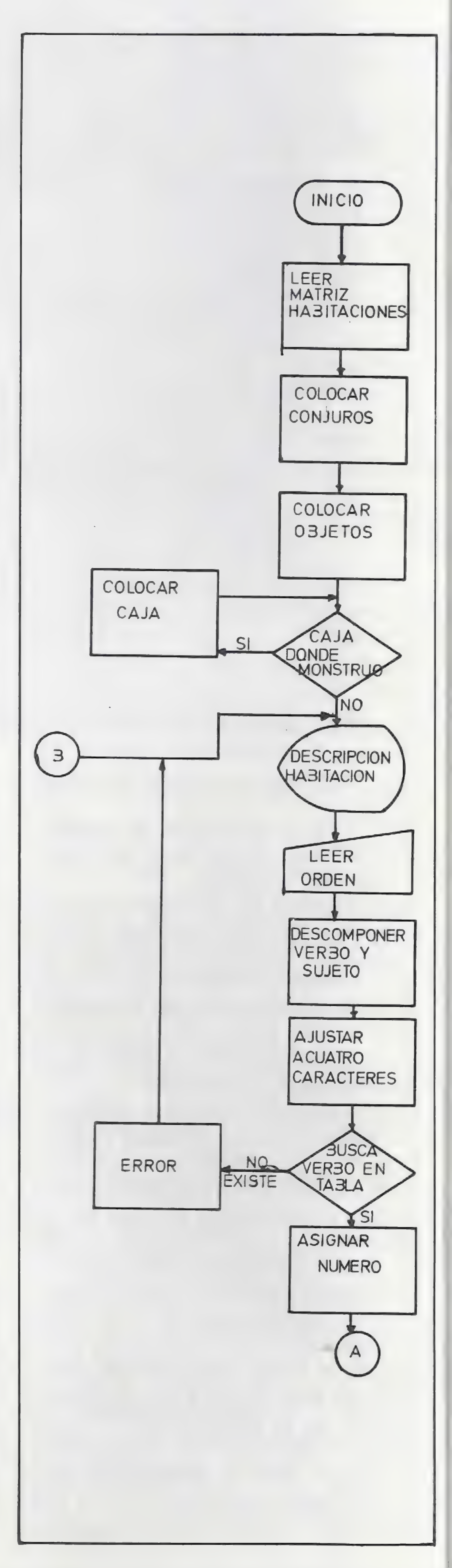
Por último, pero no por eso menos importante, debemos especificar el objetivo exacto del juego, este objetivo se le dirá al jugador de cualquier forma floreada, de modo que le dé intriga, por ejemplo, así:

"En las lejanas tierras del rey Ashglard existe un castillo abandonado hace siglos, que dicen esconde las cuatro estrellas de la inmortalidad. Muchos han sido los osados que han intentado cogerlas, pero ninguno ha vuelto, usted, caballero al servicio de su bella dama Hisdana, se las quiere ofrecer en prueba de su amor y ahora se encuentra en la puerta del castillo dispuesto a entrar a por las cuatro estrellas, que según dicen, están en un arcón encantado, que sólo se abre por medio de un conjuro." Aparte de los cursi que haya podido quedar nos da una idea muy clara de los objetivos que tenemos que alcanzar.

El organigrama básico

En la figura 2 podemos ver el esquema principal del juego en el que se pueden diferenciar varios grupos. El primero consiste en el generador del castillo, cada vez que se empiece a jugar de nuevo el programa situará los objetos en lugares distintos, de modo que sea distinto de resolver cada vez. Asímismo debe evitar los erores de bulto que pueda haber (la caja de las galletas en la misma habitación que el monstruo de las galletas). También genera dos conjuros, uno de ellos permite abrir el arcón y el otro tiene efectos bastante nocivos. La siguiente parte es el reconocedor de instrucciones, en ella se lee una orden, se codifica (dando un código inválido, si se le da una orden imposible) y se pasa al siguiente bloque del programa. A cada acción y a cada objeto se les asigna un número, ya que es más fácil para el ordenador trabajar con números que con letras. Con estos números busca en una matriz (tabla) como la indicada en la figura 3, en ella se da un código que indica si es posible o imposible. Si es imposible, se da un mensaje de error. También se da un mensaje de error si alguna de las palabras no existe. Cuando ya se ha comprobado que la instrucción tiene sentido se pasa al ejecutador de órdenes, aquí hay un bloque distinto para cada instrucción. Cada uno de ellos comprueba primero si se puede hacer (coger la caja de las galletas es posible, pero si no está en el mismo cuarto en que estamos nosotros, no se puede), luego se ejecuta y si hace falta, se ejecutan todas las acciones complementarias (si decimos el conjuro correcto, se abrirá el arcón). Por último, se comprueba si ha terminado el juego por cualquier razón, dando el resultado en caso positivo y volvien-

A
BUSCAR SUJETO EN TABLA
NO
ERROR
EXISTE
ASIGNAR NUMERO
ORDEN LOGICA
NO
ERROR
3
SI
3
ERROR
ELEGIR SUBRUTINA
HAZ MIRA ...
VE
NO
ACCION POSIBLE
SI
EJECUTAR
MUERTO
SI
MENSAJE
NO
3
NO
FUERA CASTILLO
SI
LLEVA TESORO
NO
MENSAJE
SI
MENSAJE
FIN



do al reconocedor de instrucciones en caso contrario.

El reconocedor de órdenes

Para reconocer una orden el programa debe leerla del teclado y descomponerla en verbo y sujeto. Para esto se buscan los espacios al principio y al final y se quitan. Luego se busca otro espacio en medio de la frase y se parte la frase por ese punto, por último se añaden espacios a la derecha o se quitan letras, de modo que quede con una longitud de cuatro. Este número de cuatro puede ampliarse a voluntad del programador (conviene no reducirlo), de modo que sean comprensibles todas las órdenes, en nuestro caso vale así, aunque signifique lo mismo NORT que NORTE, ya que la quinta letra es eliminada. Como parte final se busca en una tabla hasta encontrar el número asociado a esa palabra y se busca

	VE	HAZ	MIRA	COGE	DEJA	DI
MONEDA	0	0	1	1	1	0
CAJA	0	0	1	1	1	0
COLLAR	0	0	1	1	1	0
MONSTRUO	0	0	1	0	0	0
ARCON	0	0	1	1	1	0
LIBRO	0	0	1	1	1	0
REVISTA	0	0	1	1	1	0
CUADRO	0	0	1	1	1	0
INVENTARIO	0	1	0	0	0	0
CONJURO	0	0	0	0	0	1
NORTE	1	0	0	0	0	0
SUR	1	0	0	0	0	0
ESTE	1	0	0	0	0	0
OESTE	1	0	0	0	0	0
ARRIBA	1	0	0	0	0	0
ABAJO	1	0	0	0	0	0

en la tabla de la figura 3 para dar error en ese caso.

El ejecutador de órdenes

Aquí llega la instrucción ya codificada y con la lógica comprobada. Dependiendo de la acción que le hayamos dado irá a un bloque en el cual se ejecutará la orden. En caso de que sea una acción referente a objetos (COGE) comprueba que el objeto está en la misma habitación que nosotros y lo pasa a nuestra propiedad. Para hacerlo hay una matriz en la que se almacena el lugar donde está cada objeto. Un número del uno al 12 indica una habitación y el cero es que está en nuestra propiedad. Otro grupo de instrucciones necesita que tengamos el objeto con nosotros (MIRA DEJA), el tercero no necesita este tipo de condiciones (HAZ INVENTARIO, DI CONJURO) y el cuarto se refiere a movimiento (VE), por lo que comprueba que hay puerta en esa dirección y que se puede pasar.

```
10 REM ** DIMENSIONADO DE MATRICES **
20 DIM OB(8,2), HA(12,5),AC$(6),VA(16,6),O$(16)
30 REM ** GENERACION DE LA CASA **
40 REM ** COLOCAR PUERTAS **
50 FOR I=1 TO 12: FOR J=1 TO 5:READ HA(I,J): NEXT J:NEXT I
60 REM ** LEER NOMBRES OBJETOS **
70 FOR I=1 TO 16 :READ O$(I):NEXT I
80 REM ** DISTRIBUIR CONJUROS **
90 CB$="SHUM SHUM":CM$="ATKA BISKA"
100 IF RND(1)>.5 THEN CB$="ATKA BISKA":CM$="SHUM SHUM"
110 L=INT(RND(1)*3)+6
120 OB(L,2)=1
130 L=INT(RND(1)*3)+6
140 IF OB(L,2)<>0 GOTO 130
150 OB(L,2)=2
160 REM ** DISTRIBUIR OBJETOS **
170 FOR I=1 TO 8
180 L= INT(RND(1)*12)+1
190 OB(I,1)=L
200 NEXT I
210 REM ** COMPROBAR QUE MONSTRUO NO ESTE CON GALLETAS **
220 IF OB(3,1)=OB(5,1) THEN OB(5,1)=INT(RND(1)*12)+1:GOTO 220
230 REM ** LEER ACCIONES **
240 FOR I=1 TO 6
250 READ AC$(I)
260 NEXT I
270 REM ** LEER COMBINACIONES VALIDAS **
280 FOR I=1 TO 16:FOR J=1 TO 6
290 READ VA(I,J)
300 NEXT J: NEXT I
```

```
310 REM ** OBJETOS LLEVADOS CERO **
320 OL=0
330 REM ** HABITACION INICIAL 1  **
340 HJ=1
350 REM ** TESORO LLEVADO NO **
360 TE=0
370 REM ** ESTA MUERTO NO **
380 MU=0
500 REM ** RECONOCEDOR ORDENES **
510 PRINT "ESTAS EN UNA HABITACION"
520 FOR I=1 TO 8
530 IF HJ=OB(I,1) THEN PRINT "HAY ";O$(I)
540 NEXT I
550 PRINT "HAY SALIDAS A: "
560 FOR I=1 TO 4
570 IF HA(HJ,I)<>0 THEN PRINT O$(I+10)
580 NEXT I
590 IF (HA(HJ,5)<>0) AND (HJ<7) THEN PRINT "ARRIBA"
600 IF (HA(HJ,5)<>0) AND (HJ>6) THEN PRINT "ABAJO"
610 INPUT "ORDEN: ";A$
620 IF RIGHT$(A$,1)<>" " GOTO 640
630 A$=LEFT$(A$,LEN(A$)-1):GOTO 620
640 IF LEFT$(A$,1)<>" " GOTO 660
650 A$=RIGHT$(A$,LEN(A$)-1):GOTO 640
660 FOR I=1 TO LEN(A$)
670 IF MID$(A$,I,1)=" " THEN VE$=LEFT$(A$,I-1):SU$=RIGHT$(A$,LEN(A$)-I):GOTO 680
675 NEXT I
680 REM ** PONER EN CUATRO CARACTERES **
690 IF LEN(VE$)>4 THEN VE$=LEFT$(VE$,4)
700 IF LEN(VE$)<4 THEN VE$=VE$+" ":GOTO 700
710 IF LEN(SU$)>4 THEN SU$=LEFT$(SU$,4)
720 IF LEN(SU$)<4 THEN SU$=SU$+" ":GOTO 720
730 REM ** BUSCAR CODIGO VERBO **
740 FOR I=1 TO 6
750 IF VE$=LEFT$(AC$(I),4) THEN NA=I:GOTO 780
760 NEXT I
770 PRINT "NO ENTIENDO LO QUE DICE":GOTO 20000
780 REM ** BUSCAR CODIGO SUJETO **
790 FOR I=1 TO 16
800 IF SU$=LEFT$(O$(I),4) THEN NS=I:GOTO 830
810 NEXT I
820 PRINT "NO ENTIENDO LO QUE DICE":GOTO 20000
830 REM ** COMPROBAR ACCION LOGICA **
840 IF VA(NS,NA)=0 THEN PRINT" NO PUEDO HACER MILAGROS, LO SIENTO":GOTO 20000
850 REM ** EJECUTAR ORDEN **
860 ON NA GOTO 900,1000,1100,1200,1300,1400
900 REM ** IR EN DIRECCION **
905 IF (NS=16) AND (HJ<>12) THEN PRINT "NO PUEDO":GOTO 20000
907 IF (NS=16) AND (HJ=12) THEN HJ=6:GOTO 20000
910 IF HA(HJ,NS-10)=0 THEN PRINT "HAY UN PARED, NO PUEDO":GOTO 20000
915 IF HA(HJ,NS-10)=13 GOTO 930
920 MM=0:HJ=HA(HJ,NS-10):GOTO 20000
930 IF OB(1,1)=0 THEN HJ=13:GOTO 20000
940 PRINT "LA PUERTA ESTA CERRADA ":GOTO 20000
1000 REM ** HACER INVENTARIO **
1010 IF OL=0THEN PRINT "NO LLEVAS NADA":GOTO 20000
1020 FOR I=1 TO 8
1030 IF OB(I,1)=0 THEN PRINT "LLEVAS ";O$(I)
1040 NEXT I
1050 GOTO 20000
1100 REM ** MIRAR OBJETO **
1110 IF (OB(NS,2)=0) AND (OB(NS,1)=NJ)THEN PRINT " NADA ESPECIAL":GOTO 20000
1120 IF (OB(NS,2)=1) AND (OB(NS,1)=NJ)THEN PRINT CB$  :GOTO 20000
1130 IF (OB(NS,2)=2) AND (OB(NS,1)=NJ)THEN PRINT CM$  :GOTO 20000
```

```
1140 PRINT "ME RESULTA IMPOSIBLE":GOTO 20000
1200 REM ** COGER OBJETO **
1210 IF OL=2 THEN PRINT "VAS DEMASIADO CARGADO":GOTO 20000
1220 IF OB(NS,1)<>HJ THEN PRINT "NO LO VEO":GOTO 20000
1225 OL=OL+1
1230 OB(NS,1)=0:GOTO 20000
1300 REM ** DEJAR OBJETO **
1310 IF OB(NS,1)<>0 THEN PRINT "NO LO LLEVO ENCIMA":GOTO 20000
1315 OL=OL-1
1320 OB(NS,1)=HJ:GOTO 20000
1400 REM ** DECIR CONJURO **
1410 INPUT "TECLEE EL CONJURO :";A$
1420 IF (A$=CM$) AND (OB(4,1)=0) THEN PRINT "TE HA DADO UN MAREO": GOTO 20000
1430 IF (A$=CM$) AND (OB(4,1)<>0) THEN MU=1: GOTO 20000
1440 IF (A$=CB$) AND (OB(2,1)<>0) THEN PRINT "NO PASA NADA":GOTO 20000
1450 IF (A$=CB$) AND (OB(2,1)=0) THEN PRINT "TIENES EL TESORO":TE=1: GOTO 20000
1460 PRINT "APARTE DE UNA RONQUERA NO CAMBIA NADA":GOTO 20000
20000 REM ** ACCIONES SECUNDARIAS **
20010 IF HJ<>13 GOTO 20050
20020 PRINT "SALIO DE LA CASA"
20030 IF TE=1 THEN PRINT "CON LAS CUATRO ESTRELLAS. MUY BIEN":END
20040 END
20050 IF MU=1 THEN PRINT "LO SIENTO, HA MUERTO":END
20060 IF OB(3,1)=HJ THEN MM=MM+1
20070 IF (MM=2)AND(OB(5,1)<>HJ) THEN PRINT "LE COMIO EL MONSTRUO":END
20080 GOTO 500
60000 REM ** DATOS DE LAS PUERTAS **
60010 REM 1= PUERTA 0= PARED 2= PUERTA CON LLAVE
60020 DATA 3,13,0,2,0,4,0,1,0,0,5,1,0,0,0,6,2,0,0,0,3,0,6,0,0,4,5,0,12
60030 DATA 9,0,0,8,0,10,0,7,0,0,11,7,0,0,0,12,8,0,0,0,9,0,12,0,0,10,11,0,6
60040 DATA "MONEDA","ARCON","MONSTRUO","COLLAR","CAJA","REVISTA","LIBRO"
60050 DATA "CUADRO","INVENTARIO","CONJURO","NORTE","SUR ","ESTE","OESTE"
60060 DATA "ARRIBA","ABAJO"
60070 DATA "VE  ","HAZ ","MIRA","COGE","DEJA","DI  "
60080 REM ** COMBINACIONES VALIDAS **
60090 DATA 0,0,1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,1,1,1,0,0,0,1,1,1,0
60100 DATA 0,0,1,1,1,0,0,0,1,1,1,0,0,0,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,1
60110 DATA 1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0
60120 DATA 1,0,0,0,0,0
```

Todos los datos que hemos dado hasta ahora se unen en el listado del programa (figura 4). Estas reglas, como cualquier tipo de juego, no son inamovibles, sino que se pueden modificar a gusto del programador. Los datos y explicaciones que hemos dado anteriormente pueden servir de referencia para programadores que los modifiquen a su gusto.

Los juegos de aventuras en general

Los juegos de este tipo se pueden situar en cualquier lugar y ocasión. La única limitación radica en su imaginación para crearse lugares donde pelear o buscar objetos, o cualquier otra misión que se le ocurra. A continuación damos unos cuantos lugares típicos de aventura y las cosas que puede haber en ellos.

Una aventura de la Edad Media

En este tipo de aventuras abundan los monstruos en sus más diversas formas: lobos, enanos, orcos, serpientes, vampiros, inspectores de hacienda, fantasmas, etc., también hay conjuros, armas, tesoros y otro tipo de cosas habituales en el pasado.

El detective en una casa

En este caso podemos buscar un asesino o unas joyas, a nuestro alcance están todas las facilidades de la vida moderna, cerillas, lámparas, neveras, etc. Suele haber asesinos horribles que nos lanzan cuchillos desde las esquinas.

Aventuras en el tiempo

Aquí disponemos de una máquina del tiempo para trasladarnos por diversas épocas, en este caso los objetos que puede haber sólo están limitados a la imaginación (quién sabe qué nos deparará el futuro, aparte de más impuestos).

Aventura en la isla

Las ideas son las mismas, con la única limitación impuesta por la época en la que queramos ambientar el hecho.



Test

Test es un juego para el **VIC 20**, que nos envía **Alfredo José Díaz** desde Las Palmas de Gran Canaria. Es un juego en el que deben dispararse proyectiles, desde una base submarina, sobre diferentes blancos que discurren por la parte superior de la pantalla. Estos blancos incluyen helicópteros, lanchas de superficie y submarinos y cada uno de ellos puntúa de forma diferente cuando es alcanzado. Además los submarinos van soltando minas, sobre las que no hay que disparar pues restan puntos. En la parte inferior de la pantalla puede verse en cualquier momento el número de proyectiles de que se dispone para el combate, acabados los cuales termina el juego.

Al comenzar aparecen las instrucciones del juego junto con la posibilidad de escoger entre dos niveles de dificultad.

```
0 GOSUB1585
1 POKE36879,29:PRINT"█████████████FACIL=  /F/"
2 PRINT"█████████DIFICIL=  /D/"
3 P=PEEK(197):IFP=18THENOO=0:GOTO9
5 IFP=42THENOO=1:GOTO9
7 GOTO3
9 PRINT"█"
10 POKE36869,255:POKE52,28:POKE56,28
11 IFOO=1THENCLR:OO=1
12 IFOO=0THENCLR:OO=0
13 POKE36878,10
20 A=7168:FORI=ATOA+511:POKEI,PEEK(I+25600):NEXT
30 FORL=0TO111:READF:POKEA+L,F:NEXT
40 DATA1,0,0,32,119,32,0,0
45 DATA0,3,7,15,255,114,63,31
50 DATA3,3,3,127,255,213,255,127
55 DATA255,16,56,100,230,124,40,124
60 DATA0,192,224,240,255,78,252,248
65 DATA128,128,128,254,255,171,255,254
70 DATA16,16,16,16,56,254,254,254,0,16,16,0,0,16,16,0
75 DATA0,82,62,252,63,122,54,16,128,81,42,4,0,0,0,0,0
77 DATA1,130,64,28,62,28,0,0,16,16,16,56,56,56,0
79 DATA0,0,3,127,7,63,21,31,0,8,144,160,192,252,88,240
80 Z=8109:S=7834:Q=0:II=8186
85 FORR=38818TO38839:POKER,2:NEXT
87 FORR=8120TO8186:POKER,11:POKE30720+R,3:NEXT
90 FORR=7856TO7877:POKER,9:POKE30720+R,6:NEXT
100 P=PEEK(197):IFP=31THENZ=Z-1:GOTO110
105 IFP=23THENZ=Z+1
110 D=1:IFZ<8099ORZ>8118THEN200
115 IFZ1<>ZTHENQ=0:POKEZ1,32
120 POKEZ,6
122 Z1=Z
125 IFQ>2THEN200
130 IFP=8THEN:D=2:Q=Q+1:GOTO600
200 ONDIGOTO250,300,400
250 POKES,32:S=S+M:POKES+30720,4
260 IFS=7834ORS=7855THENPOKES,32:GOTO500
280 POKES,1+F1:POKES+M,1+F2:POKES+M+30720,4
290 D1=1:GOTO450
300 POKES,32:S=S+M:POKES+30720,2
310 IFS=7878ORS=7899THENPOKES,32:GOTO500
320 POKES,2+F1:POKES+M,2+F2:POKES+M+30720,2
350 IFS=CTHENPOKEC-22,10
360 D1=2:GOTO450
400 POKE36877,200:POKES,32:POKES+1,32:S=S+1
410 IFS=7833THENPOKE36877,0:GOTO500
420 POKES,0:POKES+1,3:D1=3:POKE36877,0:POKES+30721,7
```

PREMIADO CON 5.000 PESETAS

```
450 ONDGOTO100,610
500 N=RND(1):M=INT(RND(1)+.5)
510 IFM=0THENM=-1:F1=3:F2=0:GOTO530
520 F1=0:F2=3
525 IFN<.25THEND1=1:S=7854:M=-1:F1=12:F2=11:GOTO450
530 IFN<.5THEND1=1:S=7845-9*M:GOTO450
540 IFN<.75THEND1=2:S=7889-9*M:C=Z-220:GOTO450
550 D1=3:S=7812:M=1:GOTO450
600 R=Z:D=2:IFII>8186THENII=8186
605 II=II-1:IFII<8120THEN1500
607 POKEIII,32
610 FORTR=0TO00:POKE36877,220:IFR<7900THEN650
620 R=R-22
630 POKER,7:POKER+39720,7:POKE36877,0:D=2
640 NEXT:GOTO200
650 SI=R-S:SE=R-S-M
655 IFPEEK(Z-242)=10THENSC=SC-15:POKEII,11:II=II+1:POKEZ-242,8:GOSUB845
660 IF22*INT(SI/22)=SITHENSA=PEEK(S):GOSUB800:GOTO680
670 IF22*INT(SE/22)=SETHENSA=PEEK(S+M):GOSUB800
680 FORR=Z-22TOZ-220STEP-22
690 POKER,32:NEXT:POKE36877,0:GOTO100
800 IFPEEK(Z-242)=10AND(SA=20ORSA=5)THENSC=SC+80
810 IFSA=20ORSA=5THENSC=SC+200:GOSUB950:GOTO840
815 IFSA=1ORSA=4ORSA=12ORSA=13THENSC=SC+70:GOTO840
820 IFSA=0ORSA=3THENSC=SC+10
840 POKES,8:POKES+30720,7:POKE36878,15
845 FORL=0TO5:POKE36875,220:GOSUB900:POKE36875,0:GOSUB900:NEXT
850 POKEZ-242,9:POKE36878,10:GOTO1000
900 FORA=1TO90:NEXT:RETURN
950 IFD1=2THENFORR=7856TO7877:POKER,9:NEXT
960 RETURN
1000 FORA=7717TO7723:POKEA,32:NEXT
1010 SC$=STR$(SC):L=LEN(SC$)
1020 FORA=1TOL
1030 POKE7717+A,ASC(MID$(SC$,A,1))+128
1040 POKE38438+A,0:NEXT:RETURN
1500 PRINT"□":POKE36869,240
1510 PRINT"QQQQQQQQQTU TANTEO = "SC
1520 IFSC>1500THENPRINT"■QQQQ■■■■+66 MUNICIONES":FORX=0TO4000:NEXT:RESTORE:GOTO9
1550 PRINT"■■■QQQQ■OTRA VEZ ?■"
1560 GETA$:IFA$=""THEN1560
1570 IFA$="S"THENRUN
1575 IFA$<>"N"THENGOTO1560
1580 END
1585 POKE36879,29
1587 PRINT"□AYUDA A DESTRUIR LA FLOTA ENEMIGA"
1588 PRINT
1590 PRINT"CRSR↑=IZQUIERDA"
1595 PRINT"←=FUEGO"
1600 PRINT"CRSR←=DERECHA"
1605 PRINT
1610 PRINT"ELICOPTERO=10 P"
1615 PRINT
1620 PRINT"BARCOS=10 O 70 P"
1625 PRINT
1630 PRINT"SUBMARINOS=200 P"
1635 PRINT
1640 PRINT"MINAS=RESTA 15 P"
1645 PRINT
1650 PRINT"APRIETA UNA TECLA PARA CONTINUAR"
1660 GETA$:IFA$=""THEN1660
1670 RETURN
```

PROGRAMA commodore Magazine COMPROBADO

Programas

Newton

El juego muestra una gran estructura a modo de andamio, por la cual se puede mover un robot que recoge el máximo número de manzanas posibles que caen del "cielo". Pero no todo acaba ahí, porque por cada manzana que el robot no pueda recoger y que toque la estructura, supondrá un grave daño para ésta y consiguientemente entorpecerá los movimientos de tu robot. La estructura está rodeada de un gas azul, letal para el robot, por tanto si éste toca cualquiera de los bordes de la pantalla o bien se cae al suelo, queda destruido.

Ahora bien, el juego no es algo tan sencillo como recoger manzanas. Para obtener puntos deberás depositar, cuando lo estimes oportuno, las manzanas recogidas en unas cestas recolectoras situadas a ambos lados de la estructura; en su parte más alta. No todo van a ser dificultades... para cruzar de un lado a otro la pantalla —en un abrir y cerrar de ojos— se encuentran situados dos "transportadores de alta velocidad" a ambos extremos de la estructura (en la pantalla los verás justo abajo de las escaleras que llevan a las cestas recolectoras).

Como podrás observar, debes manejar el robot con gran habilidad para sortear escaleras, saltar agujeros procurando no caer ni en una zona sin salida ni al suelo, recoger manzanas y por fin depositarlas... ¡Todo un trabajo!

Para poder jugar debes disponer de Joystick, que colocarás en el PORT 2, y para hacer saltar al robot se debe pulsar el botón de disparo en coordinación con el movimiento del Joystick.

— El juego terminará cuando tu robot sea destruido, por tocar la zona azul, o bien cuando 5 manzanas caigan al suelo.

— También se han de depositar las manzanas recogidas en las cestas cuando se tengan 10 o menos.

— El nivel de dificultad, que más bien debiera llamarse de "facilidad", es posible elegirlo antes de comenzar a jugar, haciendo que las manzanas caigan más deprisa, según el número elegido (¡Ojo! el 1 es el más rápido).

CBM 64

```
0 REM***NEWTON***
5 PRINT"█████████████████████***NEWTON***"
6 PRINT"████████QUIERES INSTRUCCIONES? (S/N)"
7 GET QQ$:IFQQ$<>"S"ANDQQ$<>"N" THEN7
8 IFQQ$="N"THENPRINT"██                                        ESPERA":GOSUB1500
10 IFQQ$="S"THENGOSUB700
20 G=INT(1028+31*RND(0)):PRINT"█                    ":IFAP>10THENAP=10
21 IFLI>4THEN470
30 FORZZ=1TOQQ:POKEM,6:POKEM+CO,7:JV=PEEK(56320):FR=JVAND16:JV=15-(JVAND15)
35 POKE54296,15:POKE54277,190:POKE54276,17:POKE54278,190
40 IFJV<>0THENPOKEM,A:POKEM+CO,B:POKE54273,20:IFA=7ORA=8THENPOKEM,32
50 IFJV=4ANDFR=16THENM=M-1:IFPEEK(M)=0ORPEEK(M)=1THENM=M+1
60 IFJV=8ANDFR=16THENM=M+1:IFPEEK(M)=0ORPEEK(M)=1THENM=M-1
70 IFJV=1ANDPEEK(M)=2THENM=M-40
80 IFJV=2ANDPEEK(M+40)=2THENM=M+40
90 IFJV<>0THENA=PEEK(M):B=0:IFA<>32THENB=PEEK(M+CO)
100 IFJV=4ANDFR<>16THENGOSUB240
110 IFJV=8ANDFR<>16THENGOSUB270
120 IFPEEK(M+40)=32THENPOKEM,32:M=M+40:POKEM,6:POKEM+CO,7:GOTO120
130 IFA=5THENGOSUB300
135 IFM=1665ORM=1662THENGOSUB400
140 IFPEEK(M+40)=8ORA=8THEN500
145 POKE54273,0:NEXTZZ
150 POKEG,32:G=G+40:IFPEEK(G)=6THENAP=AP+1:POKE54273,40:A=32:POKEM,32:GOTO20
160 IFPEEK(G)<>32THENPOKEG,32:POKEG+40,32:GOSUB190:GOTO20
170 IFG>1943THENLI=LI+1:GOTO20
180 POKEG,7:POKEG+CO,10:PRINT"█"SC;" : ";AP;"  ":GOTO30
190 POKE54296,15:POKE54277,190:POKE54276,33:POKE54278,190:POKE54273,10:RETURN
200 A=PEEK(M):B=0:IFA=32THENRETURN
210 B=PEEK(M+CO):RETURN
240 IFPEEK(M-41)=0ORPEEK(M-41)=1THENRETURN
245 POKEM,A:POKEM+CO,B:M=M-41:GOSUB200:POKEM,6:POKEM+CO,7
250 IFPEEK(M+39)=0ORPEEK(M+39)=1THENRETURN
260 POKEM,A:POKEM+CO,B:M=M+39:GOSUB200:POKEM,6:POKEM+CO,7:RETURN
270 IFPEEK(M-39)=0ORPEEK(M-39)=1THENRETURN
275 POKEM,A:POKEM+CO,B:M=M-39:GOSUB200:POKEM,6:POKEM+CO,7
280 IFPEEK(M+41)=0ORPEEK(M+41)=1THENRETURN
```

```
290 POKEM,A:POKEM+CO,B:M=M+41:GOSUB200:POKEM,6:POKEM+CO,7:RETURN
300 IFAP<1THENRETURN
310 POKE53280,10:FORI=1TOAP:AP=AP-1:SC=SC+1:NEXT:POKE53280,11:RETURN
400 IFM=1665THENM=1661:POKE1665,8:POKE1665+CO,8
410 IFM=1662THENM=1666:POKE1662,8:POKE1662+CO,8
420 POKE54277,190:POKE54276,33:POKE54278,190
430 FORZ=0TO40STEP.5:POKE54273,Z:NEXTZ
440 POKE54277,0:POKE54276,0:POKE54278,0:POKE54273,0
450 A=2:B=0:RETURN
470 PRINT"□":POKE53280,0:POKE53281,0:POKE53272,21
480 PRINT"■■■■■■■■■■■■■          5:MANZANAS HAN TOCADO EL"
490 PRINT"■             SUELO DE LA PANTALLA"
495 FORI=1TO1000:NEXTI
500 POKE54277,190:POKE54276,129:POKE54278,190:POKE54273,6
510 FORX=15TO0STEP-.3:FORZ=1TO20:NEXTZ:POKE54296,X:NEXTX
520 POKE54277,0:POKE54276,0:POKE54278,0:POKE54273,0
530 PRINT"□":POKE53280,0:POKE53281,0:POKE53272,21
540 PRINT"■■■■           CONSEGUISTE■";SC;"■MANZANAS"
550 PRINT"■■■■■■■■■                 OTRO JUEGO(S/N)?"
560 GETP$
570 IFP$="S"THENGOSUB1640:GOTO20
580 IFP$="N"THENPRINT"□HASTA LUEGO.....":END
590 GOTO560
700 PRINT"□":POKE53280,0:POKE53281,0:PRINT"■                  ■■NEWTON■"
710 PRINT"■■■   ■EL OBJETO DEL JUEGO ES PROTEGER A"
720 PRINT"LAS ESTRUCTURAS DEL ANDAMIO MEDIANTE EL"
730 PRINT"MANEJO DE TU ■RECOGEDOR AUTOMATICO DE"
740 PRINT"MANZANAS■(R.A.M.)POR TODA LA ESTRUCTURA,"
750 PRINT"COLOCANDOLO DEBAJO DE LAS MANZANAS QUE"
760 PRINT"CAEN AL SUELO PARA PODER COGERLAS"
770 PRINT"■■■   UNA VEZ QUE HAYA AGOTADO SU"
780 PRINT"CAPACIDAD (10 O MENOS) DEBERA IR A"
790 PRINT"DEPOSITARLAS EN LA CESTA DE RECOGIDA EN"
800 PRINT"□LA PARTE SUPERIOR DEL ANDAMIO EN CADA  "
810 PRINT"LADO DE LA PANTALLA.PERO SI ALGUNA"
820 PRINT"MANZANA SE LE ESCAPA A TU 'R.A.M.' Y"
830 PRINT"TOCA LA ESTRUCTURA,CAUSARA IMPORTANTES"
840 PRINT"DESTROZOS,PONIENDOTELO MAS DIFICIL"
845 PRINT"■■■      ■PULSA UNA TECLA PARA CONTINUAR"
846 IFPEEK(197)=64THEN846
850 PRINT"□■■  SI LA ESTRUCTURA ESTA DETERIORADA DE"
860 PRINT"TAL FORMA QUE SEA IMPOSIBLE MOVERSE DE"
870 PRINT"UN LADO A OTRO POR LOS ANDAMIOS,ENTONCES"
880 PRINT"□TE VERAS OBLIGADO A USAR LA NAVE"
890 PRINT"TRANSPORTADORA (■ ■) CON LA CUAL TE"
900 PRINT"PODRAS MOVER DE UN EXTREMO A OTRO DE"
910 PRINT"FORMA INSTANTANEA.NO OBSTANTE,PUEDES      "
920 PRINT"□USAR ESTE METODO CUANDO QUIERAS."
940 PRINT"■■   ■EL DISPOSITIVO ANTIRROBO QUE RODEA LA"
950 PRINT"□ESTRUCTURA (AZUL■) ES LETAL PARA TU"
960 PRINT"ROBOT, POR TANTO SI SE CAE A TRAVES DEL"
970 PRINT"ANDAMIO AL SUELO,O SE VA DEMASIADO"
980 PRINT"LEJOS AL DEPOSITAR LAS MANZANAS:■■■■■■■■■■■■■■■■■■■■■■■■ ■SE DESTRUYE!!"
985 PRINT"■■CUIDADO:■SI ■5■ MANZANAS CAEN AL SUELO..."
986 PRINT"■                 ■QUEDAS DESTRUIDO!!!"
990 PRINT"■■      ■PULSA UNA TECLA PARA SEGUIR"
1000 IFPEEK(197)=64THEN1000
1010 PRINT"□■■  LA PUNTUACION LA VERAS EN LA ESQUINA"
1020 PRINT"SUPERIOR IZQUIERDA DE LA PANTALLA CON"
1030 PRINT"EL NUMERO DE MANZANAS QUE LLEVA EL"
```

Programas

Viene de la página anterior

```
1040 PRINT"ROBOT-(PUNTOS:CARGA);TU PUNTUACION FINAL"
1050 PRINT"]ES EL NUMERO DE MANZANAS DEPOSITADAS EN"
1060 PRINT"LAS CESTAS RECOLECTORAS DURANTE EL JUEGO"
1070 PRINT"■               ■MOVIMIENTOS"
1080 PRINT"■USA EL JOYSTICK EN EL PORT 2 PARA "
1081 PRINT"MOVERTE HACIA:"
1090 PRINT"   ARRIBA:ABAJO:IZQUIERDA:DERECHA."
1100 PRINT"■   ]EL DISPARADOR HACE QUE EL ROBOT SALTE"
1110 PRINT"]POR ENCIMA DE UN AGUJERO EN AQUELLA"
1120 PRINT"DIRECCION EN LA QUE SE ESTE MOVIENDO"
1140 PRINT"■■          ■■BUENA SUERTE!!"
1500 POKE56334,PEEK(56334)AND254:POKE1,PEEK(1)AND251
1510 FORI=0TO511:POKEI+12288,PEEK(I+53248):NEXT
1520 POKE1,PEEK(1)OR4:POKE56334,PEEK(56334)OR1
1530 FORCHAR=0TO8
1540 FORBYTE=0TO7:READN:POKE12288+(8*CHAR)+BYTE,N:NEXTBYTE:NEXTCHAR
1550 DATA 255,255,255,255,255,255,255,255
1560 DATA 129,66,36,24,24,36,66,255
1570 DATA 255,129,129,129,129,129,129,129
1580 DATA 192,192,192,192,192,192,192,192
1590 DATA 3,3,3,3,3,3,3,3
1600 DATA 255,255,66,102,36,60,24,24
1610 DATA 129,129,66,231,153,153,102,66
1620 DATA 0,36,126,126,126,126,60,0
1630 DATA 255,255,255,255,255,255,255,255
1640 PRINT"■■■■■■■NIVEL DE DIFICULTAD? (1-4)"
1641 GETQQ:IFQQ<1ORQQ>4THEN1641
1642 PRINT"■■■  PULSA UNA TECLA PARA EMPEZAR A JUGAR"
1650 IFPEEK(197)=64THEN1650
1700 POKE53272,29:PRINT"]":POKE53280,11:POKE53281,11:PRINT"■■■■":CO=54272
1710 PRINT" ■E                                                E"
1720 PRINT" ■@■B                                              ■B■@"
1730 PRINT" ■A■B                                              ■B■A"
1740 PRINT"  ■B                                              ■B"
1750 PRINT"  ■B                                              ■B"
1760 PRINT"  ■B                                              ■B"
1770 PRINT"  ■B                                              ■B"
1780 PRINT"  ■B                                              ■B"
1790 PRINT"  ■B      B■@@@ @@@@@ @@■B                 ■@@■B B"
1800 PRINT"  ■B      B■AAA AAAAA AA■B          B■@@■B ■AA■B B"
1810 PRINT"  ■B      B       ■D          ■B■@@@@@ ■B■AA■B    B B"
1820 PRINT"  ■B      B       ■D          ■B■AAAAA ■B   B    B B]H"
1830 PRINT" ]H■B       B B■@@ @@■B■@@   ■B       /B   B■@@ ■B■@@@"
1840 PRINT" ■@@■B ■@@■B■@ ■B■AA AA■B■AA ■@@@■B ■@@@@  ■B■AA ■B■AAA"
1850 PRINT" ■AA■B ■AA■B■A ■B ■C    ■B    ■AAA■B ■AAAA  ■B    B"
1860 PRINT"   ■B    B   B ■C     ■B          B ■C       ■B    B"
1870 PRINT"   ■B    B   B ■@@@@@@@@@    ■B ■C       ■B    B"
1880 PRINT"   ■@@@@@@@@@ ■AAAAAAAA     ■@@@@@@@ @@@@@@"
1890 PRINT"   ■AAAAAAAAA ■C          ■D    ■AAAAAAA AAAAAA"
1900 PRINT"   ■C          D C        D     C     D C     D"
1910 FORI=1024TO1944STEP40:POKEI,8:POKEI+CO,6:NEXT
1920 FORI=1063TO1983STEP40:POKEI,8:POKEI+CO,6:NEXT
1930 FORI=1984TO2023:POKEI,8:POKEI+CO,6:NEXT
1940 A=32:AP=0:SC=0:M=1828:LI=0
1990 RETURN
```

Clasificaci

Normalmente, cuando se maneja un conjunto de datos es importante que se encuentren ordenados o clasificados, de esta forma es más cómodo trabajar con ellos. Pensemos sino en lo que supondría tener que buscar un número de telófono en una guía telefónica donde no existiera ningún orden y en la que, para encontrar el número de fulano que tal, hubiera que buscar página por página, entre miles de números y nombres, hasta encontrar el número buscado; esperando que, con un poco de suerte, el número no estuviera situado en la última página de la guía. Por ello es natural que una de las tareas más corrientes al trabajar con un conjunto de datos, sea la de clasificarlos, bien en orden alfabético, cuando se trata de palabras, frases o textos, o bien en orden creciente, de menor a mayor, si los datos son valores numéricos.

Ordenar un conjunto de datos es una tarea sencilla, que todo el mundo es capaz de realizar pero, cuando el conjunto incluye un número elevado de datos —pongamos más de mil datos— es una tarea sumamente laboriosa y que requiere mucho tiempo para su realización. Estas características, es decir, que la tarea se realice con unos cuantos pasos sencillos, pero repetidos, un número muy grande de veces, nos suena inmediatamente a trabajo para un ordenador y así es un efecto. Mucho se ha trabajado en este tema y son muchos los algoritmos o procedimientos estudiados, unos más sencillos de programar y no demasiado rápidos en su ejecución, otros muchos más rápidos y eficientes pero también más complejos.

En este artículo vamos a describir cuatro de estos algoritmos de clasificación, explicando su forma de actuar y acompañando esta explicación con una rutina en BASIC, que ilustre cada uno de los métodos.

La tabla 1 muestra un resumen comparativo de cuatro métodos de clasificación de datos. En ella se indica el grado de complejidad de cada método y la eficiencia del mismo, que no es más que la mayor o menor velocidad del método a la hora de ordenar el conjunto de datos. Esta eficiencia, como puede verse en la tabla, viene referida tanto para pequeñas como para grandes cantidades de datos y suele ser distinta en cada caso. Hay otro factor a tener en cuenta, a la hora de elegir entre unas y otras técnicas de clasificación, del que no hemos hablado y que es la cantidad de memoria que necesita y utiliza cada método.

En primer lugar, todos los métodos necesitan un mínimo espacio de memoria común para todos ellos, que es el necesario para almacenar los datos, desordenados en un principio, que van a ser clasificados. Normalmente, los datos se guardan en una matríz unidimensional, bien numérica, cuando los datos son números, bien de expresiones literales cuando los datos a ordenar son caracteres o cadenas de caracteres.

Algunos métodos, como el de la burbuja, no necesitan más espacio de memoria, mientras que otros métodos si lo necesitan, normalmente para almacenar de forma temporal determinadas variables auxiliares utilizadas en el proceso de clasificación. Cuando se trabaja con una gran cantidad de datos, esta necesidad de memoria puede ser el factor más

```
10 REM ****************************
15 REM *                          *
20 REM * CLASIFICACION DE DATOS *
25 REM *    PROGRAMA PRINCIPAL    *
30 REM *                          *
35 REM *    COMMODORE MAGAZINE    *
40 REM *                          *
45 REM ****************************
50 REM
55 CLR
60 PRINT"▯▮▮▮TECNICAS DE CLASIFICACION DE DATOS"
65 INPUT"▮▮▮NO.DE DATOS (5-1000) ";N
70 IF N<5 OR N>1000 THEN 65
75 N=INT(N):DIMD(N):PRINT"▮▮ESPERA..."
80 FOR J=1 TO N
85 D(J)=INT(RND(0)*3000)
90 NEXT J
95 PRINT"▯ELIGE METODO"
100 PRINT"▮ 1) BURBUJA"
105 PRINT" 2) BURBUJA MODIFICADO"
110 PRINT" 3) INSERCION"
115 PRINT" 4) RAPIDO"
120 PRINT"▮▮PULSA NUMERO DE OPCION"
125 GETA$:IFA$=""THEN125
130 A=VAL(A$):IF A<1 OR A>4 THEN125
135 TIME$="000000"
140 ON A GOSUB 1000,2000,3000,4000
145 PRINT"▯CLASIFICACION TERMINADA"
150 H$=MID$(TIME$,1,2)
155 M$=MID$(TIME$,3,2)
160 S$=MID$(TIME$,5,2)
165 PRINT"▮▮TIEMPO EMPLEADO"
170 PRINT"▮ ";H$;" HORAS ";M$;" MINUTOS ";S$;" SEGUNDOS"
175 PRINT"▮▮▮DESEAS VER LOS DATOS ORDENADOS ? (S/N)"
180 GET R$: IFR$="" THEN 180
185 IF R$<>"S" THEN55
190 FOR J=1 TO N
195 PRINT D(J)
200 NEXT J
205 PRINT"▮▮ PULSA UNA TECLA"
210 GETR$: IF R$="" THEN 210
215 GOTO 55
```

ón de datos

importante, por delante de la velocidad o eficiencia del método.

EL PROGRAMA PRINCIPAL

Antes de pasar a explicar cada una de las cuatro técnicas, vamos a referirnos al programa principal. Tiene como misión servir de soporte a las rutinas correspondientes a las cuatro técnicas y se encarga, en primer lugar, de crear un conjunto de N datos aleatorios (N puede elegirse a voluntad), que se almacenan en una matríz D (N), estos son los datos a clasificar. Después el programa presenta un menú, en el que se puede elegir entre cualquiera de las cuatro técnicas de clasificación. Al elegir una opción se produce un salto a la subrutina correspondiente a dicha opción, de la que solo se vuelve al programa principal una vez finalizada la clasificación de los datos. Además, el programa principal se encarga de medir el tiempo transcurrido desde que se produce el salto a la subrutina hasta que se vuelve de ella, mostrando entonces en pantalla, el tiempo empleado por el método. Esta medida del tiempo empleado por cada uno de los algoritmos, en llevar a cabo la clasificación, es un aspecto muy interesante del programa. Permite hacer ensayos con las distintas técnicas y con distinto número de datos y ver, en cada caso, cual técnica es la más interesante y hacer comparaciones entre ellas. En realidad, se trata de un ejemplo de experimentación matemática y de prueba de modelos con ordenador. Además, cualquier otra técnica o algoritmo que se desee probar, puede incluirse como subrutina en este programa principal. Por último, el programa principal ofrece la posibilidad de ver los datos una vez que han sido ordenados lo que, de ensayarse otros algoritmos diferentes de los presentados en este artículo, permitirá comprobar que la ordenación de los datos se ha llevado a cabo correctamente.

EL METODO DE LA BURBUJA

El método de la burbuja es un algoritmo de clasificación muy sencillo y muy elegante al mismo tiempo. Su nombre deriva del hecho de que si se observa la tabla o matríz de datos a lo largo del proceso de clasificación, se podrá comprobar como los números más pequeños se desplazan, poco a poco, hacia la parte superior de la tabla, como si se tratara de burbujas en un líquido, mientras que los números mayores, más pesados, van hundiéndose lentamente pasando a ocupar las últimas posiciones en la tabla. Parece como si la ordenación se produjera espontáneamente, por razones de densidad y peso de los números, casi como algo natural, cuando por el contrario, lo más natural es el desorden.

El algoritmo en sí consiste en lo siguiente: una tras otra, se van realizando pasadas sobre la tabla de datos, de arriba abajo. En cada pasada se trata de comparar cada número con el que está inmediatamente debajo de él y actuar con ellos de la siguiente forma: si el que está debajo es mayor, no hacer nada, pues esa pareja de números ya está ordenada, el número mayor, el que más pesa debajo y el más pequeño encima. Si por el contrario el que está debajo es

```
1000 REM METODO DE LA BURBUJA
1005 REM ********************
1010 REM
1015 PRINT"]METODO DE LA BURBUJA":PRINT"XESPERA..."
1020 FOR J=1 TO N-1
1025 F=0
1030 FOR I=1 TO N-J
1035 IF D(I)<=D(I+1) THEN 1050
1040 F=1
1045 T=D(I+1):D(I+1)=D(I):D(I)=T
1050 NEXT I
1055 IF F=0 THEN 1065
1060 NEXT J
1065 RETURN
```

```
2000 REM METODO DE LA BURBUJA MODIFICADO
2005 REM ******************************
2010 REM
2015 PRINT"]METODO DE LA BURBUJA MODIFICADO":PRINT"XESPERA..."
2020 LI=N
2025 IF LI<1 THEN 2080
2030 LI=INT(LI/2)
2035 F=0
2040 FOR I=1 TO N-LI
2045 IF D(I)<=D(I+LI) THEN 2060
2050 F=1
2055 T=D(I+LI):D(I+LI)=D(I):D(I)=T
2060 NEXT I
2065 IF F=0 THEN 2025
2070 GOTO 2035
2080 RETURN
```

Tabla 1.

METODO	COMPLEJIDAD	EFICIENCIA POCOS DATOS	EFICIENCIA MUCHOS DATOS
Burbuja	Muy simple	Excelente	Pobre
Burbuja modificado	Media	Buena	Buena
Inserción	Muy simple	Excelente	Media
Clasificación rápida.	Grande	Buena	Muy buena

Clasificación

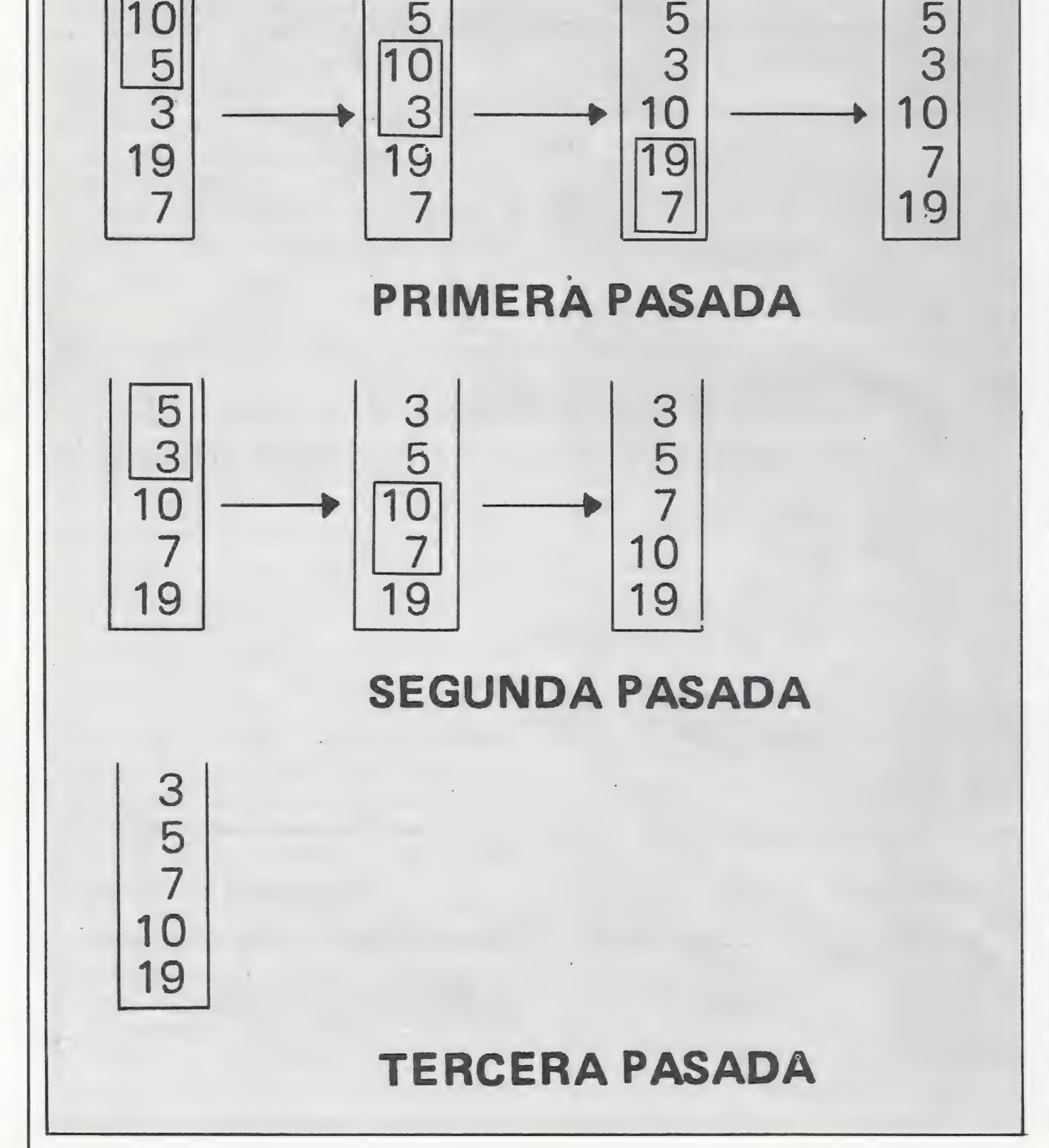
menor, hay que intercambiar la pareja de números antes de pasar a la pareja siguiente. De esta forma, en cada pasada se intercambian algunas parejas de números, que diremos que han producido una burbuja, mientras que otras parejas quedan como estaban. Es fácil ver que en el momento en que, en una pasada, no se produzca ninguna burbuja, es porque la taba de números ya está ordenada y el algoritmo ha terminado con la clasificación. Es así de sencillo. Un ejemplo de todo esto puede verse más claramente a través de los dibujos de la figura 1, donde las burbujas se han marcado con círculos y donde la clasificación se realiza completamente en 3 pasadas.

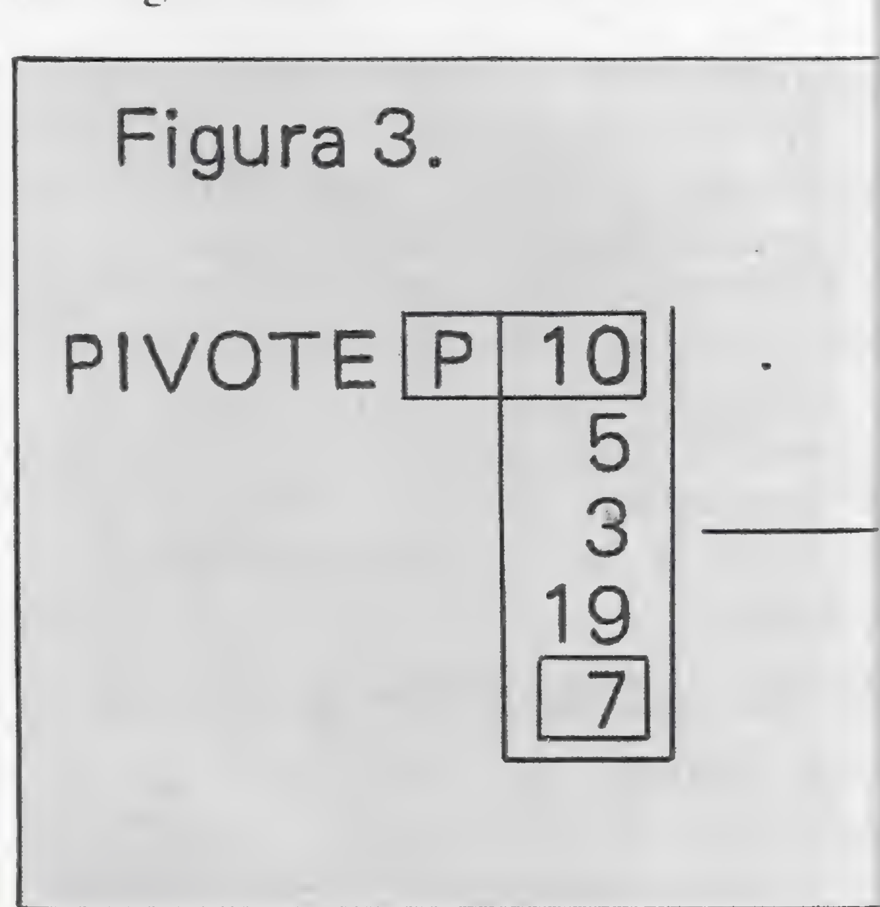
METODO DE LA BURBUJA MODIFICADO

Con el método de la burbuja, y dentro de cada pasada de exploración, se lleva a cabo una comparación entre números adyacentes, para ver si hay que intercambiarlos o no.

El método de la burbuja modificado hace lo mismo pero de otra forma, basándose en una hipótesis muy interesante que dice que es mejor comparar primero números alejados entre sí en la tabla, para, si hay que intercambiarlos, hacerlos cuanto antes. De esta forma, en el método de la burbuja modificado, se establece un "intervalo de clasificación", que no es más que la distancia, en la tabla, entre números que se van a comparar. En una primera pasada esta distancia es igual a la mitad de la longitud de la tabla, es decir, que si la tabla contiene por ejemplo, 100 datos, en la primera pasada se lleva a cabo la comparación entre datos separados 50 posiciones en la tabla. En las sucesivas pasadas el intervalo de clasificación se reduce cada vez a la mitad, es decir, que en nuestro ejemplo de 100 datos, en la segunda pasada el intervalo de clasificación valdría 25 y se llevaría a cabo la comparación entre datos separados por 25 posiciones, en la tercera pasada el intervalo valdría 12, en la cuarta 6 y así sucesivamente hasta que, en la última pasada, el intervalo valdría 1, con lo que se haría la comparación entre datos adyacentes tal y como se hace en cualquiera de las pasadas del método de la burbuja.

¿Cuál es la ventaja de este proceder que justifique la mayor complejidad del algoritmo?





de datos

La verdad es que para un pequeño número de datos, este método no presenta ninguna ventaja frente al de la burbuja, pero cuando el número de datos a clasificar crece, las cosas son muy diferentes. Si consideramos por ejeplo, una tabla con 100 datos y en la que, ¿por qué no?, el valor más pequeño, que hay que colocar en la primera posición de la tabla, se encuentra inicialmente en la última posición, es fácil comprobar que, con el método de la burbuja hay que realizar 99 pasadas sobre los datos, mientras que con el método de la burbuja modificado, el número de pasadas es de 6, independientemente de como estuvieran de desordenados los datos inicialmente. Para mayor número de datos, la diferencia es todavía mayor, y queda justificada plenamente la mayor complejidad del algoritmo, por el ahorro de tiempo que supone. Invitamos al lector a que lo compruebe por sí mismo.

EL METODO DE INSERCION

El método de inserción es sin lugar a dudas, el que guarda una mayor similitud con el algoritmo que normalmente desarrolla la mente humana, cuando se enfrenta con la tarea de tener que clasificar un conjunto de datos. Es un algoritmo muy simple de programar, muy eficiente para pequeñas cantidades de datos, pero no demasiado bueno cuando el número de los mismos se hace elevado. En esencia el algoritmo se desarrolla de la siguiente manera:

En primer lugar, todo el proceso de clasificación se lleva a cabo en una sola pasada, de arriba abajo, por la tabla. En el transcurso de esta pasada, cada número se compara con todos los que están por encima de él y se inserta en el lugar que le corresponde, debajo de todos los números menores que él y por encima de todos los que son mayores. Después, se coge el número que ocupa la siguiente posición y se repite el proceso. De esta forma, en cualquier momento, todos los números de la tabla por encima de la posición que se considera, están perfectamente ordenados. Cuando se llega al último número de la tabla, y después de insertarlo en el lugar que le corresponde, el proceso de clasificación ha terminado y la tabla está perfectamente ordenada. Todo este proceso queda mucho más claro al observar el ejemplo de la figura 2 en el que se detallan las distintas fases en la clasificación de una tabla de 5 elementos.

EL METODO DE CLASIFICACION RAPIDA

Este método es con mucho el más complejo de los métodos que se consideran en este artículo, si bien, como contrapartida lógica es el más rápido de todos ellos.

El algoritmo consiste, en esencia, en dividir la tabla de datos en varios subgrupos, a medida que se realiza una primera ordenación. Cada uno de estos subgrupos se divide nuevamente en subgrupos, los cuales se vuelven a dividir, y así sucesivamente hasta que los subgrupos son lo suficientemente pequeños como para aplicar algo parecido al método de la

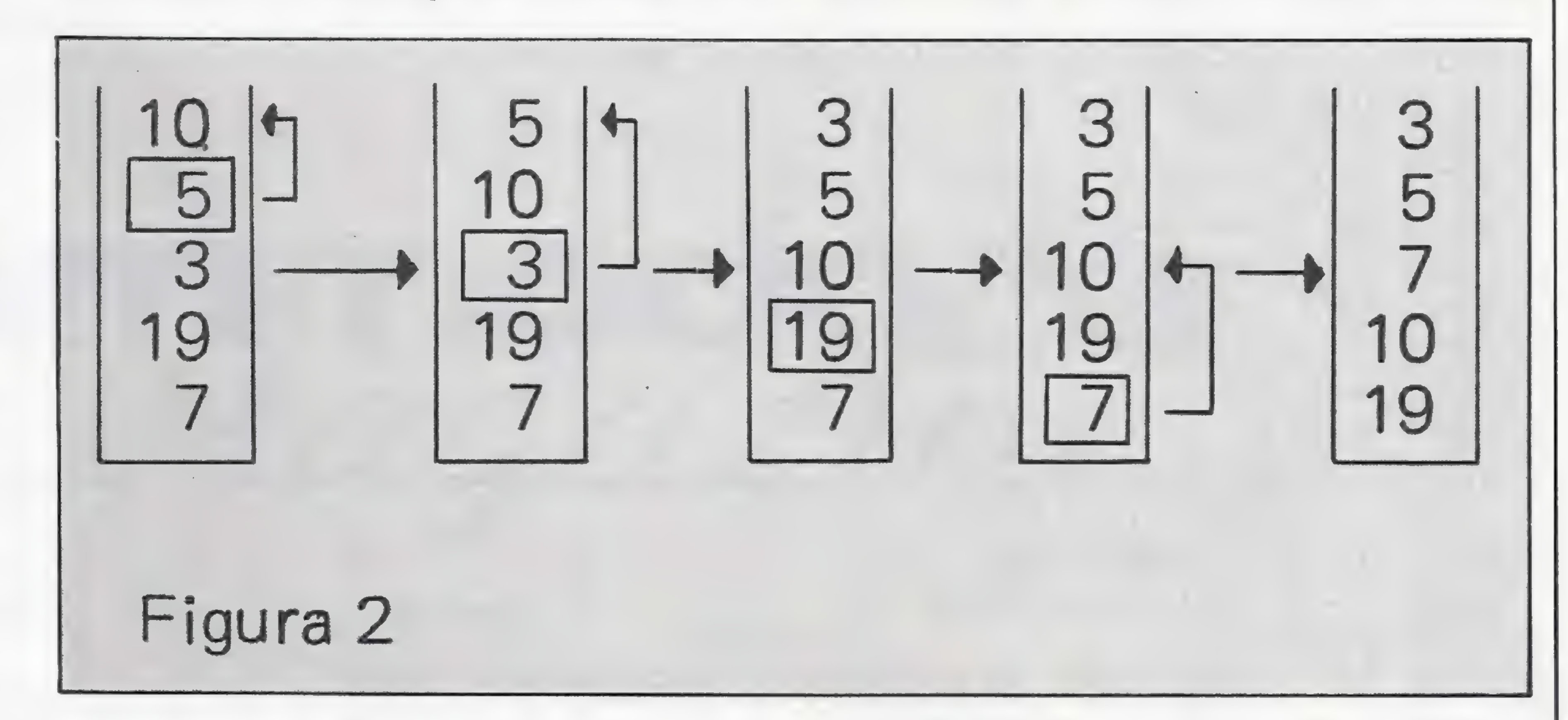
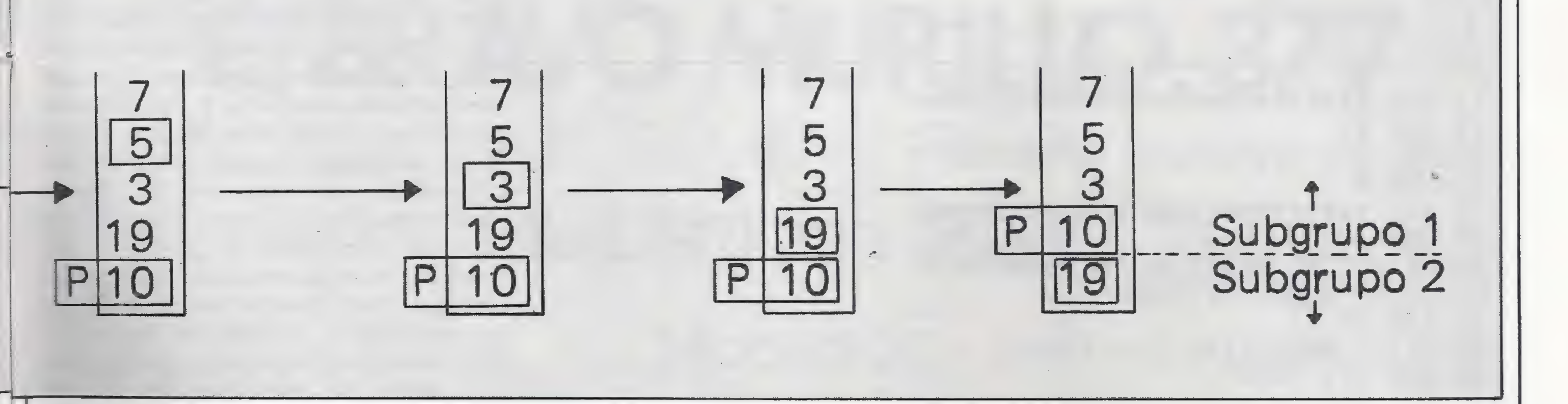


Figura 2



Clasificación

burbuja, en cada uno de ellos. Durante todo este proceso de división en subgrupos, la información sobre donde está situado cada subgrupo se almacena en una matriz adicional. Cada vez que se crea un nuevo subgrupo con ciertas condiciones, se añade un nuevo elemento a esta matriz. Al finalizar la división, se empiezan a extraer datos de esta matríz auxiliar, en el mismo orden como se introdujeron en ella, es decir, el primero en entrar es el primero en salir, y se procede a ordenar cada uno de los subgrupos. En el momento en que esta matriz auxiliar quede vacía quiere decir, que se ha terminado de ordenar el último subgrupo, con lo que el algoritmo termina.

Durante todo el proceso, el algoritmo utiliza dos punteros o indicadores de posición para moverse por la matriz. Los dos valores señalados por estos punteros se intercambian si el que se encuentra más arriba en la tabla es mayor que el que está más abajo, dejándose como estaban en caso contrario. Uno de los punteros, que se denomina pivote, señaa siempre hacia el elemento que ocupaba la primera posición de la matríz, mientras que el otro puntero se mueve por la matríz hacia el pivote. Cuando ambos se encuentran, la matríz se divide en dos subgrupos y el proceso comienza de nuevo en cada subgrupo, almacenándose la información necesaria en la matríz auxiliar. En caso de trabajar con un número muy elevado de datos, puede ocurrir que la matríz auxiliar se quede pequeña, por lo que la rutina lleva previsto un mensaje de desbordamiento de la matríz auxiliar, para esta eventualidad. Para aclarar un poco las ideas, hemos preparado el dibujo de la figura 3 donde se detalla el proceso de subdivisión en subgrupos.

```
3000 REM METODO DE INSERCION
3005 REM ******************
3010 REM
3015 PRINT"METODO DE INSERCION":PRINT"ESPERA..."
3020 FOR J=2 TO N
3025 T=D(J)
3030 FOR I=J-1 TO 1 STEP-1
3035 IF D(I)<=T THEN 3050
3040 D(I+1)=D(I)
3045 NEXTI
3050 D(I+1)=T
3055 NEXT J
3060 RETURN
```

```
4000 REM METODO RAPIDO
4005 REM ************
4010 REM
4015 PRINT"METODO RAPIDO":PRINT"ESPERA..."
4020 DIM S(40,2)
4025 PS=1
4030 S(1,1)=1:S(1,2)=N
4035 IF PS=0 THEN4115
4040 II=S(PS,1):JJ=S(PS,2):PS=PS-1
4045 P=0:I=II:J=JJ
4050 IF D(I)<D(J) THEN4065
4055 P=1-P
4060 T=D(I):D(I)=D(J):D(J)=T
4065 IFP=0THENI=I+1
4070 IFP=1THENJ=J-1
4075 IF I<J THEN4050
4080 IF I>=JJ THEN 4100
4085 PS=PS+1
4090 IF PS>40 THEN PRINT"PILA DESBORDADA":END
4095 S(PS,1)=I+1:S(PS,2)=JJ
4100 JJ=I-1
4105 IF JJ>II THEN4045
4110 GOTO 4035
4115 RETURN
```

```
10 REM *****************************
15 REM *                           *
20 REM * CLASIFICACION ALFABETICA *
25 REM *                           *
30 REM *****************************
35 REM
40 PRINT"CLASIFICACION ALFABETICA"
45 INPUT"NO. DE FRASES";N
50 DIM A$(N)
55 PRINT"ESCRIBA CADA FRASE"
60 FOR J=1 TO N
```

CLASIFICACION ALFABETICA

Hasta ahora sólo hemos hablado de la clasificación u ordenación de valores numéricos, pasando por alto el importante aspecto de la clasificación alfabética de palabras, frases y textos. Sin embargo, en este apartado es válido todo lo dicho hasta ahora: los métodos de la burbuja, burbuja modificado, inserción y clasificación rápida son todos ellos válidos, con la única diferencia de que tanto la matríz donde se almacean los datos, como los propios datos, dejan de ser valores numéricos y pasan a ser cadenas de caracteres, por lo que para trabajar con ellos es necesario añadir el símbolo $ a final de todas las variables utilizadas en el programa. La razón de que esto sea así de

de datos

sencillo y no sea necesario realizar más modificaciones, estriba en que el BASIC de **Commodore** considera una cadena de caracteres mayor que otra, cuando va después en orden alabético. Por ejemplo, la cadena A$ = "BUENO" es "mayor" que la cadena B$ = "BUENA" ya que va después en orden alfabético, pero es "menor" que la cadena C$ = "BUENOS". Esta propiedad permite utilizar cualquier algoritmo de clasificación tanto para valores numéricos como para cadenas de caracteres, sin más que sustituir las variables numéricas que manejan datos numéricos, por variables de caracteres terminadas en $. También hay que tener en cuenta que la matríz donde se almacenan los datos deja de ser una matríz numérica y que también hay que añadirle al final el carácter $.

Como ejemplo de todo esto presentamos el programa CLASIFICACION ALFABETICA que utiliza el método de inserción para ordenar alfabéticamente un conjunto de cadenas de caracteres.

```
65 INPUT A$(J)
70 NEXT J
75 FOR J=2 TO N
80 T$=A$(J)
85 FOR K=J-1 TO 1 STEP-1
90 IF A$(K)<T$ THEN 105
95 A$(K+1)=A$(K)
100 NEXT K
105 A$(K+1)=T$
110 NEXT J
115 PRINT"██LISTA ORDENADA█"
120 FOR J=1 TO N
125 PRINT A$(J)
130 NEXT J
135 PRINT"█PULSA ESPACIO"
140 GET K$:IF K$=" " THEN 40
145 CLR:GOTO140
```



Dibuja lo que quieras

Es posible crear visualizaciones tremendamente buenas utilizando únicamente bloques gráficos predefinidos, una característica de la que disfrutan también los ordenadores de **Commodore.** Esto es suficiene durante algún tiempo. Sin embargo, llega el momento en el que deseamos ampliar los horizontes de las posibilidades gráficas de nuestro ordenador. Lograr definir caracteres propios es una forma de conseguirlo.

Tanto el **Vic 20** como el **CBM 64** llevan incorporado, en su memoria ROM, un conjunto de patrones en forma de bits que conforman los caracteres visualizables, que por otra parte no pueden ser alterados. No obstante, al contrario de lo que ocurre con otros sistemas, permiten depositar copia de esta información en la RAM, lo cual posibilita que los caracteres puedan ser alterados.

Los principios de este método son idénticos tanto para el **64** como para el **Vic 20.**

Para comenzar hay que advertirle al ordenador que consiga la información de video a partir de la RAM, en lugar de la ROM. Cargando la dirección decimal de memoria 36869, en el **Vic 20,** o la 53272, en el **64** con un determinado número mediante un POKE, se logra este fin. El número dependerá de en que lugar de la RAM se desee almacenar los caracteres.

A continuación se debe proporcionar la información a la máquina. Se vuelve a acometer la tarea utilizando POKE, para depositar información en determinadas posiciones de la memoria. Si deseamos mantener determinados caracteres previamente existentes (por ejemplo, las letras del alfabeto), simplemente hay que copiarlos a partir de la ROM. EL cuadro 1 muestra cómo encontrarlo.

La creación de nuevos caracteres requiere en primer lugar que sean diseñados sobre papel, calculando después sus valores que los definen. La explicación de cómo se acomete esta parte queda mejor explicada en el cuadro 2.

Dada la pequeña cantidad de memoria que tiene el **Vic** sin expansión, merece la pena tener en cuenta que cuando se envían caracteres a la RAM se está consumiendo espacio que podría ser utilizado por un programa.

Cada carácter del juego completo necesita ocho bytes de memoria para quedar definido. El juego completo viene a ocupar unos cuatro Kbytes.

Se podría derivar alguna complicación del hecho que tanto el **Video Interface Chip (VIC)** como el **VIC II,** pueden solamente "ver" 16 Kbytes de la memoria al mismo tiempo, por lo que es necesario mantener la memoria destinada al contenido de la pantalla y el juego de caracteres dentro del mismo bloque de 16 Kbytes. En las guías de referencia para el programador se puede ampliar esta información.

MODOS GRAFICOS

Los modelos **Vic 20** y **CBM 64** ofrecen una amplia variedad de modos gráficos, siendo más completa la correspondiente al segundo. El modo que vemos más a menudo es el texto, por ser el estándar que aparece cada vez que conectamos el ordenador. Esto viene a significar que podemos acceder directamente a todos los caracteres impresos en el teclado, por medio de las teclas SHIFT y CTRL.

Se puede visualizar texto en cualquiera de los ocho colores de que dispone el **Vic 20,** o de los 16 del **CBM 64.**

Utilizando POKE en una de las localizaciones del **VIC,** se puede entrar en el modo multicolor, lo cual nos permite disponer de más de dos colores dentro de la celdilla correspondiente a cada carácter (fondo y primer plano). En este modo podemos tener reborde, fondo, primer plano y colores auxiliares.

El incremento en la resolución en colores se consigue a costa de la resolución en *pixels* (cada punto independiente de otro que aparece en la pantalla). Esto se debe a que cada



punto pasa a tener el doble de anchura que previamente.

El modo multicolor cobra su razón de ser cuando se utilizan caracteres propios.

El **CBM 64** dispone de otro modo más, el modo fondo extendido. Con él se puede establecer individualmente el color de fondo de cada carácter, además de tener el color de pantalla normal.

COLOR

El color puede añadir mucho atractivo a los programas, tanto si son juegos como algún otro *software* de índole más práctica. Por ejemplo, podemos emplear diferentes colores para resaltar el texto en bases de datos o tratamientos de texto.

Se podría alterar el color del reborde para significar algo al usuario, por ejemplo verde para las entradas de datos del usuario, azul para indicar que el ordenador está ocupado, o rojo como advertencia de que se está escribiendo en la cinta o el *diskette* y podrían quedar inutilizados.

Además de la pantalla por mapa de memoria (significa que cada posición destinada a caracteres tiene su homóloga en una dirección de memoria), el **Vic 20** y el **CBM 64** tienen pantallas por mapa de color. En el **Vic 20** ésta comienza en la dirección 38400, en el modelo sin expansión, y en la 37888 en la versión expandida. En el **64**, la primer localización es la 55296.

Introduciendo un número entre 0 y 8 en el **Vic 20**, o entre el 0 y el 16 en el **CBM 64**, mediante POKE dentro de la memoria del color es posible controlar directamente el color de cualquiera de los caracteres que aparecen en una posición de la pantalla.

El movimiento de objetos sobre la superficie de la pantalla en color es sencillo, aunque algo más laborioso. Los números clave a utilizar son el comienzo de la memoria destinada a la pantalla, el comienzo de la memoria destinada al color, la longitud de la línea y la longitud de la página.

Introducir un objeto en la memoria



destinada a la pantalla, utilizando POKE más la longitud de la línea, hará que el objeto aparezca en ella. Para crear el efecto de movimiento, simplemente habrá que dejar que el carácter aparezca, esperar un ratito, hacerlo desaparecer visualmente y después volverlo a poner en otro lugar.

Para hacerlo que baje una línea a partir de la posición inicial requiere que se sume la longitud de una línea a esa posición. Para moverlo hacia arriba se sustrae la longitud de la línea. Moverlo a la izquierda o la derecha es necesario sumar o restar uno. El movimiento en diagonal son combinaciones sencillas de estas cuatro posibilidades.

Algo a tener en cuenta: si los caracteres están en el extremo derecho de la pantalla, moverlo otra vez a la derecha lo desplazará al extremo izquierdo, pero una línea más abajo. Esta característica llamada enrollamiento podría ser útil en algún caso.

	Vic20	CBM64
Mayusculas, graficos	32768	53248
Mayusculas, graficos en inverso	33792	54272
Mayusculas y minusculas	34816	55296
Mayusculas y minusculas en inverso	35840	56320

Cuadro 1. Localización de los juegos de caracteres en la ROM.

Valor del bit	7 128	6 64	5 32	4 16	3 8	2 4	1 2	0 1	Total
No. de linea 1				*	*				24
2			*	*	*	*			60
3		*	*			*	*		102
4	*		*			*		*	165
5	*		*	*	*	*		*	189
6			*	*	*	*			60
7		*					*		66
8		*					*		66

Cuadro 2. Creación de caracteres propios.

Este invasor marciano puede ser dibujado en un papel cuadriculado en formato de 8 × 8 puntos. Para determinar el valor de cada byte que compone el carácter, solamente hay que sumar al total el valor correspondiente a cada asterisco en cada línea. Al final obtenemos ocho números comprendidos entre cero (ningún punto en la línea) y 255 (todos los puntos de la línea). Si se introdujeran estos valores en la memoria mediante POKE, como segundo carácter propio, el invasor aparecería en la pantalla cada vez que se presione la tecla A (el primero sería @).

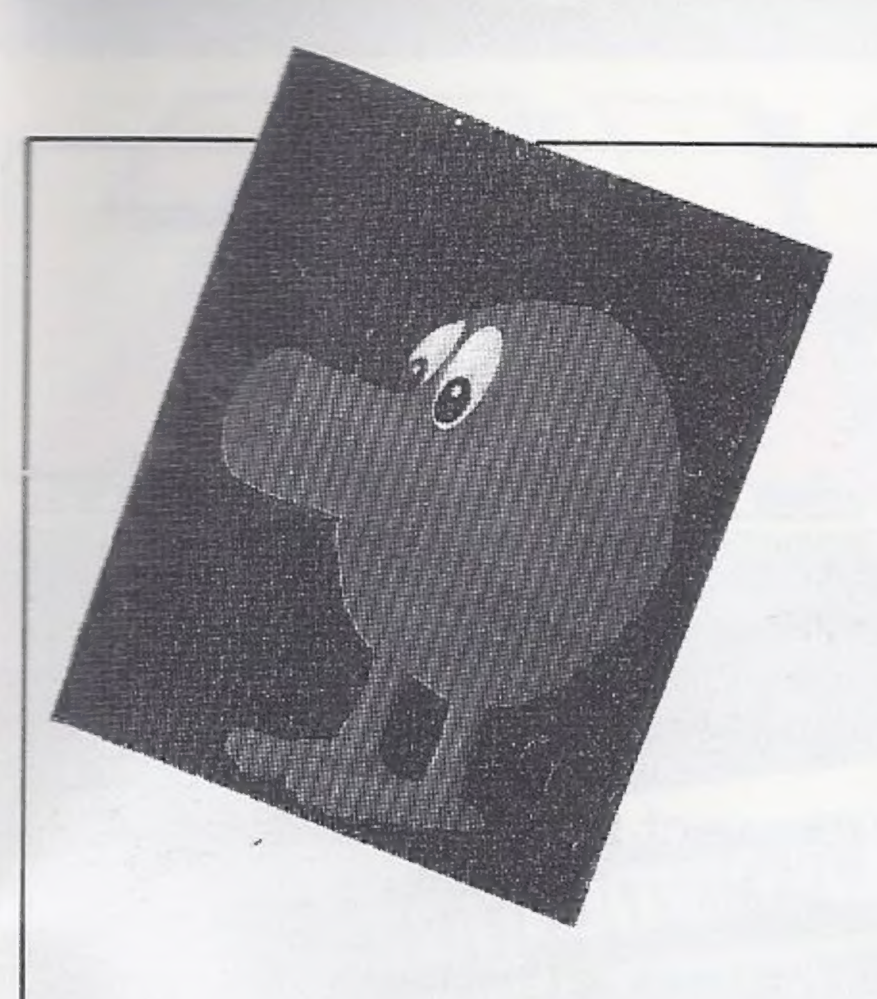
ALTA RESOLUCION

Aunque el tipo de animación previamente descrito es simple y puede ser efectivo, tiene un pequeño defecto, que es que la unidad de distancia más pequeña utilizable es la correspondiente al tamaño de un carácter, es decir, ocho puntos de una vez. La solución se encuentra en los gráficos de alta resolución, lo cual implica que la mínima distancia para el desplazamiento es de un punto. La utilización de este modo nos permite conseguir visualizaciones con alto grado de detalle y movimiento más elegante.

Existen dos obstáculos. El primero es que este tipo de animación consume grandes cantidades de memoria y, el segundo, es que **Commodore** no ha incluido comandos en el BASIC de ninguno de los dos ordenadores que permitan la utilización directa de este método.

Encontrar la razón del enorme consumo de cantidad de memoria requerida es una simple cuestión de matemáticas. Se necesita un byte para visualizar una línea compuesta por ocho puntos. Puesto que el **Vic 20** tiene un formato de pantalla de 22 líneas de ancho y 184 líneas de "profundidad", se necesitan unos 4 Kbytes de memoria para que la pantalla quede totalmente comprendida en un mapa de memoria. Esta cifra se incrementa a 8 Kbytes con el **64**, por disponer de una pantalla de mayor resolución.

El segundo problema puede ser solventado en una de dos maneras posibles. Podemos escribir programas propios para obtener visualizaciones en alta resolución, o podríamos ad-



quirir uno de esos programas de ayuda a la elaboración de gráficos. Los gráficos pueden ser confeccionados a partir del BASIC, pero la principal dificultad se centrará en la velocidad. La alta resolución implica cientos de cálculos repetitivos, que serán extremadamente lentos en el caso del BASIC, por lo que la necesidad real se centra en el código máquina.

Los gráficos son marginalmente más sencillos en el **CBM 64** por dos razones principales: la primera es que internamente dispone de facilidades orientadas a la utilización de la alta resolución, por las cuales podemos establecer nuestras propias necesidades de visualización. Segunda, si lo que necesitamos es un "*scrolling* fino" (desplazamiento del contenido de la pantalla en cualquier sentido, como cuando las líneas de texto desaparecen por arriba apareciendo nuevas por debajo —*scrolling* horizontal— de manera cuasi continua a la vista). Pues bien, este tipo de *scrolling*, destinado a la animación, se consigue mediante POKE aplicados a un par de registros internos y una corta rutina en código máquina.

53248.	Posicion horizontal del Sprite 0.
53249.	Posiscion vertical del Sprite 0.
53250-53263.	Igua que los anteriores, pero para los Sprites 1 a 7.
53264.	Posicion horizontal del Sprite mayor de 255 (ver la nota).
53265.	Registro de scrolling vertical.
53266.	Registto de barrido.
53267.	Posicion horizontal del lapiz optico.
53268.	Posicion vertical del lapiz optico.
53269.	Activa y desactiva los Sprites.
53270.	Registro de scrolling horizontal.
53271.	Expande los Sprites en sentido vertical.
53272.	La mitad alta del byte establece la direccion de la pantalla. La mitad baja establece la direccion de la memoria de caracteres.
53273.	Registro de peticion de interrupcion.
53274.	Registro de mascaras de interrupcion.
53275.	Establece la prioridad entre Sprites y fondo.
53276.	Selecciona Sprites multicolor.
53277.	Expande los Sprites horizontalmente.
53278.	Detecta la colision entre Sprites.
53279.	Detecta la colision entre Sprite y fondo.
53280.	Establece el color del reborde.
53281.	Establece el color de la pantalla.
53282.	Primer color del fondo.
53283.	Segundo color del fondo.
53284.	Tercer color del fondo.
53285.	Color cero del Sprite, en modo multicolor.
53286.	Color uno del Sprite, en modo multicolor.
53287-53294.	Establece el color de los Sprites cero a siete.

Cuadro 4. Registros del *chip* VIC II del CBM 64.

Nota: Debido a que el máximo valor de cualquier registro sólo puede ser 255, y las coordenadas de la pantalla en alta resolución pueden llegar hasta 320, se requiere un registro adicional para mantener la posición horizontal de los *Sprites*. Estableciendo el bit relevante de este registro, el VIC II cuenta las posiciones por encima de 255 como posición cero en adelante.

SPRITES

Los *Sprites* son esos pequeños duendes que podemos definir en la pantalla y que pueden moverse haciendo honor a su nombre. Este es el punto donde se divorcian las capacidades del **Vic 20** y el **CBM 64**. Desafortunadamente para los usuarios del **Vic** no son tan simples para este modelo.

En esencia los *Sprites* no son mucha cosa más que grandes caracteres definidos por el usuario. Es más, son creados exactamente de la misma manera.

La inmensa utilidad de los *Sprites* para el programador de gráficos se debe a las capacidades del *chip* **VIC II**. Una serie de registros nos permite efectuar un *scrolling* fino (contrario a *scrolling* discreto, o a saltos) del *Sprite* en cualquier dirección, simplemente utilizando POKE.

Podemos cambiar de color, emplear el modo multicolor, detectar colosiones entre *Sprites* o establecer prioridades para crear la ilusión de cuál queda en primer término con respecto a los demás.

Todas estas características hacen que la creación de visualizaciones atractivas sea extremadamente sencilla y muy rápida.

Simultáneamente, es posible ejercer control sobre hasta ocho *Sprites*,



teniendo cada uno su propio color y recorriendo cada uno su camino por la pantalla de modo independiente a los demás.

Las técnicas para todo este control salen fuera del objetivo del presente artículo. Por necesitar de más espacio, lo trataremos más adelante en sucesivos números en cada uno de sus detalles. De todas maneras existen libros excelentes, aunque la mayoría en inglés, que tratan el tema con gran dedicación.

Cuando se comprende con extensión el funcionamiento y control de los *Sprites,* despegarse del teclado del ordenador exige un esfuerzo titánico.

36864.	Los bits 0 a 6 controlan el centrado horizontal de la imagen. El bit 7 establece el entrelazado de lineas del barrido de la imagen en pantalla.
36865.	Controla el centrado vertical de la imagen.
36866.	Los bits 0 a 6 determinan el numero de columnas en la pantalla. El bit 7 es parte de la direccion de la matriz de video.
36867.	Los bits 1 a 6 determinan el numero de filas en la pantalla. El bit 0 establece si los caracteres son normales o de doble altura.
36868.	Registro para el barrido en la pantalla del televisor.
36869.	Los bits 0 a 3 establecen el comienzo de la memoria de caracteres. Los bits 4 a 7 son el resto de las direcciones de la matriz de video.

Cuadro 3. Localizaciones del *chip* VIC para el Vic 20.